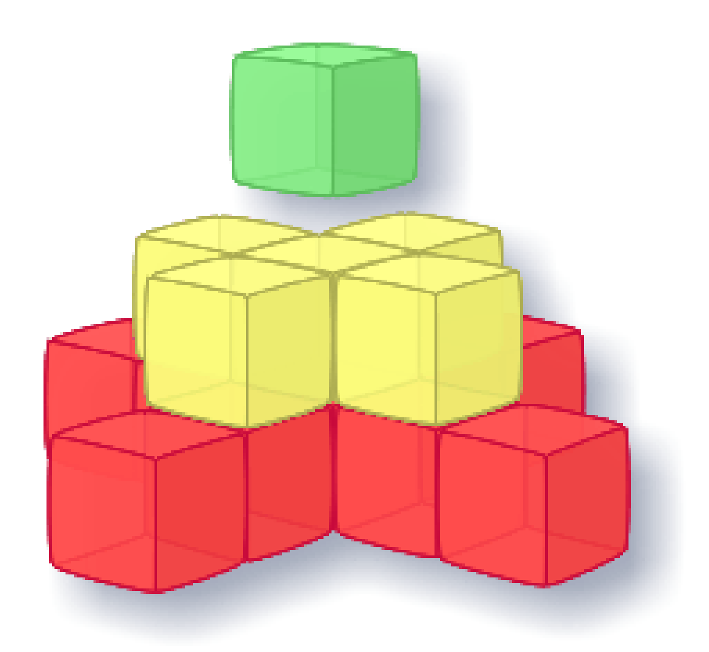
Dipl. Ing. Toralf Riedel
Dipl. Ing. Päd. Alexander Huwaldt

Benutzerhandbuch

SiSy® AVR und AVR ++

gültig ab SiSy-Version 3.10



7. Auflage: Mai 2011

www.laser-co.de
www.myavr.de
service@myavr.de
Tel: ++49 (0) 3585 470 222
Fax: ++49 (0) 3585 470 233

Inhalt

1 Einleitung

Sie haben eine Ausgabe des Modellierungswerkzeuges Simple System, kurz SiSy, erworben. Bevor auf die verschiedenen Funktionen des Programms eingegangen wird, noch einige Worte zum vorliegenden Handbuch. Mit Hilfe des Handbuchs werden dem Nutzer die Grundlagen der Bedienung von SiSy erläutert. Der Inhalt, die Gestalt und die Regeln der Modelle werden nur am Rand betrachtet. Das genaue Vorgehen für die Programmierung eines Mikroprozessors wird an einem Beispiel ausführlich beschrieben. Auf die Grundlagen der Mikroprozessorprogrammierung wird im Rahmen dieses Handbuches nicht eingegangen. Dazu dienen die myAVR Lehrbücher.
Dem Nutzer wird in diesem Handbuch der Einstieg in das Programm erleichtert und die umfangreichen Funktionen von SiSy kurz und verständlich beschrieben. Bei der Arbeit mit SiSy erstellt der Anwender Modelle in Form von Diagrammen und in ihnen enthaltene Symbole. Die Grundlagen der Entstehung und Bearbeitung solcher Diagramme sind Gegenstand der Betrachtung dieses Handbuchs.
Folgende Darstellungs- und Gestaltungsmittel sind für den Nutzer bei der Arbeit mit SiSy besonders wichtig:

- die Diagramme als Fenster zur Ansicht und Bearbeitung von Modellen;
- der Navigator als Fenster zur Steuerung und Bewegung in Modellen;
- der Assistent mit Hilfestellungen zum jeweils geöffneten Diagramm und mit Diagrammvorlagen (wenn vorhanden);
- die Menüs und Schalter für Befehle an Navigator, Diagramm und Objekt im Kontext mit der Modellierung.

Zu den Bezeichnungen im Text:

- Falls ein Menübefehl nur über Untermenüs zu erreichen ist, werden die einzelnen Menübezeichnungen kursiv geschrieben und durch Schrägstriche voneinander getrennt.
  Beispiel: Menü *Hilfe/über SiSy*

- Titel von Dialogboxen, Schaltflächen und Menüpunkten werden in Anführungszeichen gesetzt.
  Beispiel: Dialogbox „Definition“, Schaltfläche „OK“

2 Vorbereitung

In diesem Kapitel werden Sie über notwendige Schritte zur Installation, Konfiguration und Aufbau einer funktionsfähigen Entwicklungsumgebung informiert.

2.1 Installation der Software

Für die Bearbeitung der Übungen und Aufgaben steht Ihnen die AVR-Entwicklungsumgebung SiSy AVR bzw. SiSy AVR++ zur Verfügung. Sollten Sie SiSy bereits installiert haben, können Sie dieses Kapitel überspringen.

2.1.1 Voraussetzungen

Für die Installation benötigen Sie einen Freischaltcode (Lizenzangaben). Falls Sie diese Angaben nicht mit der Software erhalten haben, können Sie diese online abrufen von

www.myAVR.de → Online-Shop→ Kontakt/Service

oder fordern Sie diese beim Hersteller an:

Tel: 03585-470222
Fax: 03585-470233
e-Mail: service@myAVR.de.

Bitte prüfen Sie, ob die Systemvoraussetzungen für die Installation und die Arbeit mit SiSy AVR gewährleistet sind.

- für das Board myAVR Board MK2 USB und myAVR Board MK3 einen PC-Arbeitsplatz oder Notebook mit USB-Anschluss
- für das myAVR Board MK1 LPT einen PC-Arbeitsplatz oder Notebook mit LPT-Port und mindestens einem COM-Port
- ab Pentium III oder äquivalent
- 800 MB freier Speicherplatz auf der Festplatte
- Windows XP, Windows Vista oder Windows 7
- 256 MB RAM
- Microsoft Internet-Explorer ab Version 7.0
- Maus oder ähnliches Zeigegerät
- Assembler Entwicklungsumgebung (in SiSy bereits integriert)
- myAVR Board
- USB Kabel (myAVR Board MK2 USB) bzw. LPT-Verlängerung (myAVR Board MK1 LPT)
- Null-Modemkabel (bei myAVR Board MK1 LPT)
- Bei Bedarf (z.B. autonomer Einsatz des myAVR Boards) geeignete Spannungsversorgung z.B. 9 V Batterie / stabilisiertes 9 V Netzteil

Des Weiteren sollten Sie Grundkenntnisse in einer beliebigen Programmiersprache besitzen. Die Installation und der erste Start müssen mit Administratorrechten durchgeführt werden.

2.1.2 Setup von der SiSy-CD

Legen Sie die CD „SiSy“ in Ihr CD-ROM-Laufwerk ein. Falls die CD nicht automatisch startet, wählen Sie bitte im Explorer das CD-ROM-Laufwerk und starten die setup.exe aus dem Pfad *CD-Laufwerk:\Ausgabe\SiSy*.
Auf dem Startbildschirm stehen Schaltflächen zur Verfügung zum Installieren der Software und zum Öffnen von Begleitdokumenten.
Für die Installation der Software betätigen Sie die entsprechende Schaltfläche. In Abhängigkeit Ihrer Rechnerkonfiguration kann der Start des Setup-Programms einige Se-

kunden dauern. Das gestartete Setup-Programm wird Sie durch die weitere Installation führen.

Beginn der Installation

Betätigen Sie im Setup-Programm die Schaltfläche „Weiter“. Sie erhalten die Lizenzbestimmungen. Bitte lesen Sie diese sorgfältig durch. Wenn Sie sich mit diesen Bestimmungen einverstanden erklären, bestätigen Sie die Lizenzbestimmungen mit der Schaltfläche „Annehmen“.

Sie werden im folgenden Dialog dazu aufgefordert, Ihre Lizenzangaben einzugeben.



Danach erscheint die Dialogbox „Komponenten auswählen“, welche Sie mit „Weiter“ bestätigen. Im darauf folgenden Fenster können Sie festlegen, unter welchem Pfad SiSy AVR installiert werden soll. Wenn ein anderer Pfad (bzw. ein anderes Laufwerk) gewünscht wird, ist die Schaltfläche „Durchsuchen“ zu betätigen. Eine Dialogbox erscheint, in der Sie Laufwerk und Verzeichnis auswählen können.

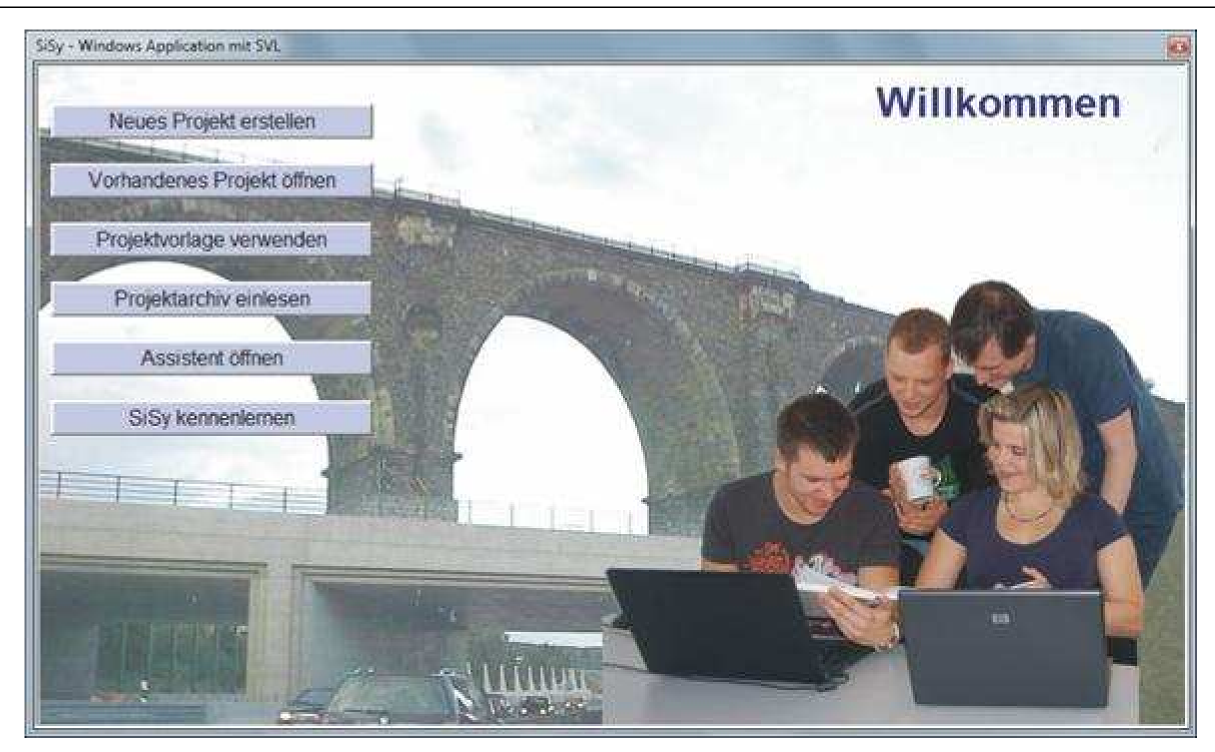




Bestimmen Sie danach den Startmenü-Ordner, in dem die Verknüpfungen von SiSy eingefügt werden. Sie können den Zielordner ändern. Sie können dies durch auswählen von „Keine Verknüpfungen erstellen“ unterbinden.

Beginnen Sie nun die Installation durch Betätigen der Schaltfläche „Installieren“. Die Installation wird nach auswählen von „Fertig stellen“ abgeschlossen.

Sie können nun SiSy AVR starten. Es erscheint auf Ihrem Bildschirm der Dialog „Willkommen in SiSy“. Folgen Sie dann den Hinweisen des Assistenten, indem Sie „Assistent öffnen auswählen“.



Hinweis:
In SiSy sind 2 Dateien enthalten, die Makros beinhalten („handbuch.doc“, „multi.doc"). Von einigen Virenscannern werden diese Makros als „Virus“ erkannt und entsprechend behandelt. In den Heuristik-Einstellungen des Virenscanners kann diese Behandlung unterdrückt werden.

2.2 Beschaffen bzw. Herstellen der Hardware

Alle Ausführungen, Übungen und Aufgabenstellungen beziehen sich auf das myAVR Board als Referenzhardware. Wenn Sie Spaß an Elektronik haben, können Sie die Hardware auch selbst fertigen. Die Komponenten erhalten Sie als Bausatz oder fertig bestückt inklusive Schaltplan etc. unter www.myAVR.de.
Ausführliche Beschreibungen zur Programmierung mit Assembler, C/C++, BASCOM sind nicht Inhalt dieses Handbuches. Weiterführende Erklärungen dazu sind im „myAVR Lehrbuch Mikrocontrollerprogrammierung“ enthalten.



myAVR Board mit LPT- bzw. USB-Port sowie das myMultiProg und der mySmart-Control

3 Arbeiten mit SiSy AVR, allgemein

Um dem Nutzer zu erläutern, wie er in SiSy modellieren kann, werden zweckentsprechende Definitionen der Begriffe gegeben.

3.1 Projektarbeit

3.1.1 Was ist ein SiSy-Projekt?

Ein SiSy-Projekt ist eine abgegrenzte Menge von verknüpften Elementen für ein zu bearbeitendes Problem. Alle aktuellen Daten sind in einer Projektdatenbank gespeichert. Die Projektdatenbank besteht aus einer Anzahl von Dateien im Projektverzeichnis, wobei jedes Projekt sein eigenes Verzeichnis hat. Durch das Anlegen eines Projektarchivs können diese in einer Datei komprimiert werden.

3.1.2 Neues Projekt erstellen

Ein neues Projekt wird eingerichtet und für die Bearbeitung bereitgestellt. Die Definition (Erstellung) eines neuen Projektes erfolgt durch Vergabe eines Projektdateinamens und/oder durch Überschreiben eines alten Projektes gleichen Namens. Über die Schaltfläche „Verzeichnis“ kann die vorgeschlagene Pfadangabe geändert werden.

Hinweis:
Falls ein bereits vorhandener Projekt- oder Verzeichnisname gewählt wird, erscheint folgende Meldung: *„Das Verzeichnis ist bereits vorhanden.*
Darin befinden sich Projektdateien.
Soll das alte Projekt überschrieben werden?“

Nach der Vergabe eines Projektnamens kann im nachfolgenden Fenster ein Vorgehensmodell ausgewählt werden. Je nach Vorgehensmodell können Vorlagen oder Assistenten das Erstellen eines neuen Projektes unterstützen.

3.1.3 Vorhandenes Projekt öffnen

Ein vorhandenes Projekt wird geöffnet und die abgespeicherten Diagramme und Objekte sowie alle Möglichkeiten für die weitere Bearbeitung werden verfügbar gemacht. Die Wahl des Projektes erfolgt durch Klicken auf den entsprechenden Namen oder über die Schaltfläche „Suchen“.

Beim Aktivieren der Checkbox „Beschreibung anzeigen“ werden zum markierten Projekt Informationen angezeigt. Dazu gehören außer der Beschreibung auch Informationen zum Bearbeitungsstand.

3.1.4 Projekt archivieren

Menü *Projekt/Archiv/Anlegen.*
Es kann ein komprimiertes Archiv des Projektes erzeugt werden. Dies ist besonders aus Gründen der Datensicherheit sinnvoll. Zielverzeichnis und Dateiname für die Archivdatei werden vorgeschlagen und können korrigiert werden. Der Umfang der Archivierung ist festzulegen und die Entscheidung für eine Komprimierung zu treffen, wobei diese empfohlen wird. Wenn ein Projekt unter einem bereits vorhandenen Archivnamen angelegt werden soll, wird eine Warnung vor dem Überschreiben angezeigt. Bei Auswahl von „Nein“ wird die Erstellung des Archivs abgebrochen, bei „Ja“ wird das Projekt archiviert.

Hinweis: SiSy bietet die Möglichkeit des regelmäßigen Abspeicherns verschiedener Arbeitsstände, d.h. ein archiviertes Projekt wird nicht überschrieben. Ein Projektstand kann in einer neuen Archivdatei abgelegt werden

3.1.5 Projektarchiv einlesen

Menü *Projekt/Archiv/Einlesen*.

Hierunter versteht man das Einlesen eines Archivs zum Zweck der Rekonstruktion des Projektes.

Einlesen bedeutet Entpacken eines archivierten Projektes. Dazu sind der Archivpfad und der Dateiname des Archivs sowie das Zielverzeichnis anzugeben.

Hinweis:

Wenn im Zielpfad des Entpackens bereits ein Projekt existiert, erscheint eine Überschreibwarnung.

3.1.6 Projektarchiv als Email versenden

Menü *Projekt/Archiv/Als Email versenden…*.

Um Projekte per Email zu versenden, sollten diese zu einem Archiv zusammengefasst werden. Die Funktion „Projektarchiv als Email versenden“ erzeugt von dem aktuell geöffneten Projekt ein Archiv, startet das Standard-Email-Programm Ihres Systems, legt eine neue Email an und fügt das Archiv als Anlage ein. Sie können jetzt die Email schreiben und sofort das Projektarchiv senden.

Hinweis:

Das Standard-Email-Programm Ihres Systems muss ordnungsgemäß eingerichtet und registriert sein, damit diese Funktion korrekt ausgeführt werden kann. Es ist nicht möglich, diese Funktion mit einem WEB-Browser-Email-Programm zu nutzen.

3.2 Die Modellierungselemente von SiSy

Werkzeug

SiSy stellt für die Bearbeitung der Modelle und Teilmodelle Werkzeuge der entsprechenden Methodik bereit. Werkzeuge sind Editoren, mit denen in einem Fenster die grafische Darstellung (Diagramme) der Modelle bearbeitet werden kann.

Modellelemente

Modelle bestehen aus Elementen, welche zueinander in Beziehung stehen. Diese Modellelemente werden nach bestimmten Regeln (Metamodell) durch SiSy verarbeitet und dargestellt (Notation). Modellelemente sind in der Regel Knoten (Objekte), Kanten (Verbindungen), Rahmen, Referenzen auf Modellelemente und Texte. Prinzipiell kann jedes Modellelement verfeinert werden. Typisch ist es Knoten (Objekte) dadurch zu verfeinern, dass „unter“ diesen Objekten weitere Modellelemente angelegt werden. Dabei entsteht eine Modellhierarchie. Jedes Modellelement verfügt über einen Kurznamen, einen Langnamen, eine Beschreibung und modellinterne Attribute (Inhalt).

Diagramme

Diagramme sind grafische Darstellungen von Modellen oder Teilmodellen, die mit einem bestimmten Werkzeug erstellt werden. Die Modellierungselemente werden als Objekte in den Diagrammen unter Einhaltung von Regeln zusammengestellt.

Objekte

Objekte sind mögliche Modellelemente in Diagrammen, z.B. „kleines Programm“ in der „Programmierung“. Objekttypen sind konkrete Ausprägungen von Objekten, die in einem Diagramm angelegt wurden, z.B. Objekttyp „Lauflicht“ vom Objekt „kleines Programm“.

Referenzen

Die Objekte eines Diagramms können in anderen Diagrammen wiederverwendet werden. Durch das Hineinziehen aus dem Navigator oder aus einem offenen Diagramm wird eine Referenz vom Originalobjekt erzeugt. Die Referenz ist nur ein Verweis auf das Original, alle angezeigten Informationen wie Kurzname, Langname und Beschreibung werden vom Original bezogen. Somit sind die Informationen in allen Referenzen eines Objektes identisch mit dem Original. Dadurch werden Änderungen dieser Informationen automatisch auf alle Referenzen übertragen. Weiterhin ist es möglich, diese Referenzierung über einen sogenannten Report auszuwerten.

Kanten

Kanten sind Verbindungselemente zwischen Objekten. Eine Verbindung wird durch Ziehen mit der Maus (linke Maustaste) vom Verteiler des selektierten Objektes auf das gewünschte Objekt erreicht. Nach Loslassen der Maustaste und Prüfung der Verbindungszulässigkeit durch SiSy erscheint ein Kanten-Dialog, in dem das Element definiert und individuelle Einstellungen getroffen werden können.

Hinweis:

Bei Verbindung mit gehaltener STRG-Taste wird die Prüfung vernachlässigt und eine Zwangsverbindung erreicht.

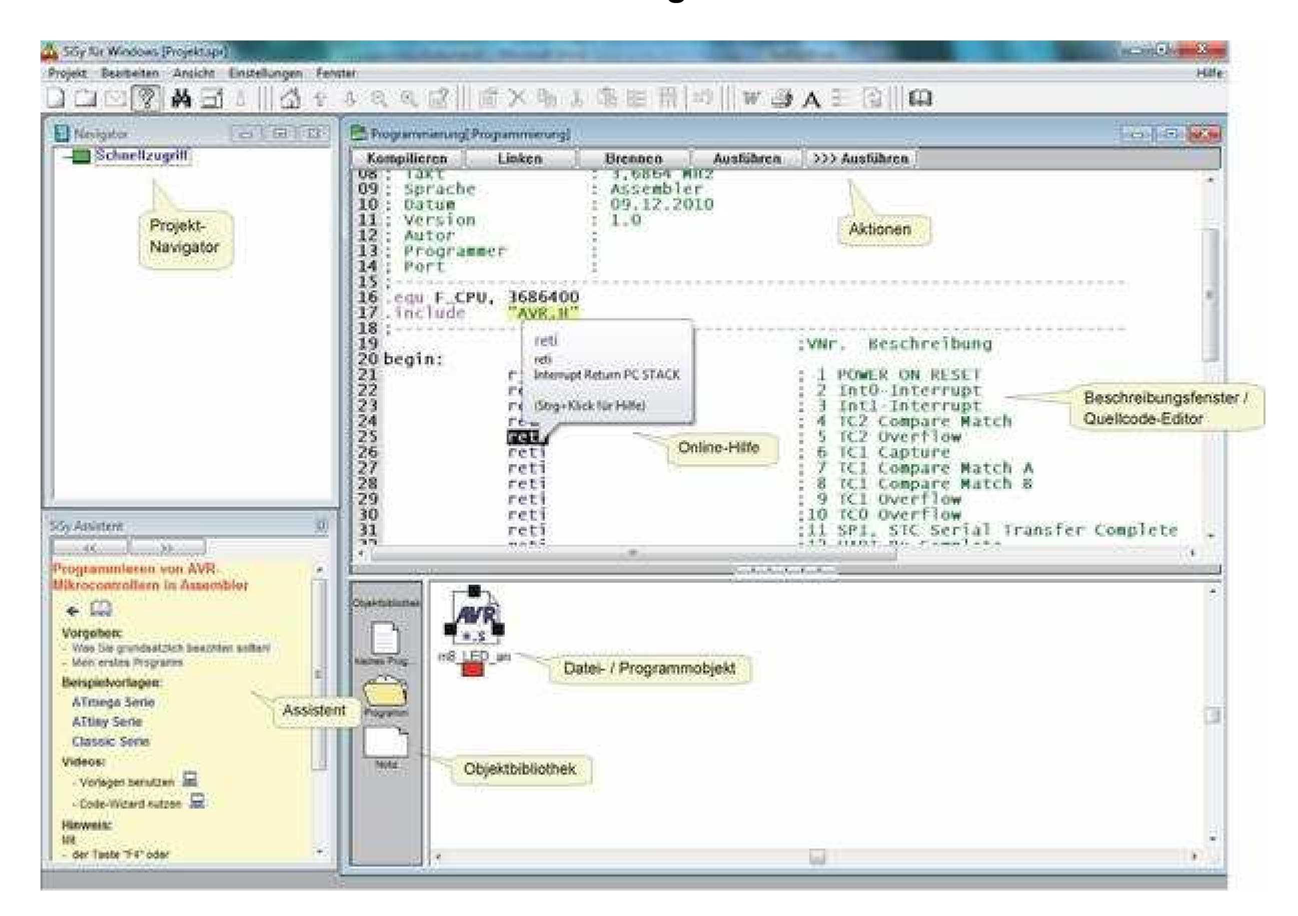
Rahmen

Ein Rahmen fasst ausgewählte Objekte des Diagramms optisch zusammen. Er besitzt einen Kurz- sowie Langnamen und eine Objektbeschreibung, kann also als Objekt aufgefasst werden.

Hinweis:

Inhalte von Rahmen sind in Reports oder einer Dokumentengenerierung auswertbar.

3.3 Die Fenster für die Modellierung



Navigator

Dieser befindet sich im linken, oberen Teil des Fensters. Er ermöglicht dem Anwender die Orientierung über die in der Projektdatenbank gespeicherten Objekte sowie deren Bereitstellung für die weitere Verwendung. Nach dem Start von SiSy werden neben dem Vorgehensmodell eine Reihe leicht zu handhabender Schaltflächen, Menüs und weitere Hilfsmittel angezeigt.

SiSy-Assistent

Am linken unteren Bildschirmrand befindet sich diese Nutzerunterstützung.

- Er begleitet den Anwender durch das gesamte Projekt und hält immer passende Informationen zur aktuellen Sicht parat
- Er ist auf die jeweilige Ausgabe von SiSy bezogen.
- Oft können Beispiele als Vorlagen vom Assistenten geladen werden.

Diagrammfenster

Am oberen Bildrand befinden sich das Hauptmenü, das hilfreiche Funktionen zu SiSy bereithält, und eine Werkzeugleiste, mit deren Hilfe schnell auf nützliche Anwendungen zugegriffen werden kann. Das Diagrammfenster nimmt die rechte Bildschirmseite ein und ist der Raum, in dem der Nutzer modelliert. Es enthält:

- das ausgewählte Vorgehensmodell,
- die Objektbibliothek mit den möglichen Objekten des aktuellen Diagramms sowie
- ein Fenster zur Beschreibung des markierten Objekts, in diesem Fall zum Editieren des Quelltextes.

Die Bedienelemente/Objektbibliothek

SiSy bietet, wie bei Windows-Anwendungen üblich, die Steuerung von Befehlen über das Hauptmenü, über die Werkzeugleisten, die Tastatur oder die Objektbibliothek an. Darüber hinaus enthalten das Kontextmenü und der Navigator Steuerfunktionen.

Die Anzahl der möglichen Befehle in der Menüleiste ist abhängig davon, ob ein Projekt geöffnet ist. Ist das nicht der Fall, erscheint ein Menü mit wenigen Befehlen. Bei einem geöffneten Projekt hält SiSy umfangreichere Menüs bereit. Die wichtigsten Menübefehle befinden sich auch als grafische Schaltfläche in der Werkzeugleiste, die eine schnelle und effiziente Bedienung des Programms ermöglicht. Die Toolbox-Darstellung bietet dem Anwender wichtige Programmfunktionen als direkten Link an.
Ein weiteres Bedienelement ist die Objektbibliothek. Sie unterstützt das Anlegen neuer Objekte.

3.4 Modelle bearbeiten

3.4.1 Modellelemente anlegen

Modellelemente können auf verschiedene Art und Weisen erzeugt werden. Der häufigste Weg ist das Anlegen über die Objektbibliothek. Dabei wird das gewünschte Element per Drag und Drop von der Objektbibliothek in das dazugehörige Diagrammfenster gezogen. Referenzen auf vorhandene Modellelemente werden üblicherweise dadurch erzeugt, dass diese per Drag und Drop aus dem Navigator in das gewünschte Diagramm gezogen werden. Verbindungen lassen sich am Einfachsten über den Verteiler herstellen. Sie können aber Modellelemente auch durch kopieren oder importieren anlegen.

3.4.2 Modellelemente auswählen

Die meisten Modellelemente lassen sich in Diagrammen auswählen (selektieren). Das Selektieren erfolgt in der Regel durch Anklicken (linke Maustaste) mit dem Mauscursor. Eine Selektion per Klick mit der rechten Maustaste ist ebenfalls möglich, dabei öffnet sich nach dem Auswählen sofort das Kontextmenü für das selektierte Modellelement. Das Betätigen der Tabulatortaste bewirkt das Auswählen des nächsten Modellelementes in der Reihenfolge des Anlegens der Elemente. Wird über die Auswahl mit der Tabulatortaste das letzte Element erreicht, beginnt die nächste Iteration wieder beim ersten Modellelement im Diagramm. Durch Betätigen der Umschalttaste kann die Iterationsrichtung umgeschaltet werden. Es ist ebenfalls möglich, mehrere Modellelemente auszuwählen. Dafür muss beim Selektieren mit der Maus die Umschalttaste gehalten werden. Bei gehaltener Umschalttaste lassen sich mehrere Objekte (keine Kanten und Rahmen) durch Ziehen einer Markise mit dem Mauscursor auswählen. Über den Menüpunkt *Bearbeiten/Diagramm/Alles auswählen* können Sie alle Objekte des aktiven Diagramms selektieren.

3.4.3 Modellelemente untersuchen (Report)

Modellelemente sind in den meisten Fällen nicht isoliert sondern haben vielfältige Beziehungen zu anderen Modellelementen. Um bei Änderungen, wie zum Beispiel Umbenennen oder Löschen, mögliche Auswirkungen auf das gesamte Modell zu überblicken, bietet der Objektreport die Möglichkeit die Modellbeziehungen des ausgewählten Objektes auszuwerten. Wählen Sie ein Objekt aus und öffnen Sie den Reportdialog über das Hauptmenü, die Werkzeugleiste oder das Kontextmenü (diagrammabhängig). Wählen Sie die erforderliche Modellanfrage und starten den Report. Das Ergebnis der Modellanfrage (z.B. Alle angebundenen Objekte) wird als Liste angezeigt. Sie können dann zu diesen Modellelementen navigieren.

3.4.4 Modellelemente verschieben

Bestimmte Modellelemente (z.B. Knoten/Objekte) lassen sich im Diagramm frei anordnen. Das Positionieren erfolgt in der Regel durch Drag und Drop mit der Maus. Die Modellelemente werden dabei an einem Raster ausgerichtet. Das Raster erleichtert die Positionierung. Um die Rasterfangfunktion zu unterbinden, muss beim Drag und Drop eines Elementes die Umschalttaste betätigt werden. Viele Modellelemente lassen sich auch über die Cursortasten verschieben. Dabei bewegen sich die Modellelemente in Richtung der betätigten Taste auf dem Raster. Auch dabei lässt sich durch die Umschalttaste die Rasterfangfunktion unterbinden. Bei Kanten wird ein Stützpunkt verschoben, der in Abhängigkeit der ausgewählten Kantenform zum gewünschten Ergebnis der Darstellung führt. Rahmen haben nur eine eigene Position, solange diese leer sind. Ein Rahmen bezieht sich in Position und Größe immer auf die darin enthaltenen Modellelemente (umschließen). Beim Verschieben eines Rahmens mit enthaltenen Elementen wird nicht der Rahmen, sondern die enthaltenen Modellelemente verschoben.

3.4.5 Die Größe von Modellelementen verändern

In machen Diagrammen verfügen Modellelemente über Anfassmarken zur Änderung der Größe. Die manuelle Größenänderung von Modellelementen ist jedoch in vielen Diagrammtypen nicht erwünscht. In diesen Diagrammen stehen keine Anfassmarken zu Größenänderung zur Verfügung. Sollte in Ausnahmefällen eine manuelle Anpassung der Objektgröße gewünscht sein, kann dies für selektierte Objekte durch die Kombination von gehaltener Steuerungstaste und den Cursortasten bzw. die Tasten Plus, Minus und Mal des Numerikfeldes erfolgen.

3.4.6 Modellelemente verbinden

Knoten und Rahmen können über Kanten verbunden werden (Verbindungen). Verbindungen werden als Linien mit unterschiedlichen Linientypen und -formen sowie Symbolen (Anfang, Ende, Mitte) und diagrammspezifischen Beschriftungen dargestellt. Das Verbinden erfolgt in der Regel über den Verteiler (rotes Rechteck an der Unterkante) eines selektierten Objektes. Dabei wird per Drag und Drop vom Verteiler die Verbindung vom Quellelement zum Zielelement gezogen. Es ist nicht nötig exakt den Rand des Zielelementes zu treffen sondern besser, die Verbindung weit in das Zielobjekt hinein zu ziehen. Nach dem Loslassen der linken Maustaste wird die Verbindung erzeugt und die Randpunkte bis zu denen die Verbindung gezeichnet wird (Clipping) automatisch ermittelt. Sie können während des Drag und Drop durch Klick mit der rechten Maustaste bereits den zukünftigen Stützpunkt der Verbindung festlegen.

3.4.7 Modellelemente löschen

Zu löschende Modellelemente müssen zuerst selektiert werden. Die Löschaufforderung erfolgt mittels der Taste Entfernen, der Werkzeugleiste oder über das Hauptmenü *Bearbeiten/Objekt/Löschen* bzw. das Kontextmenü. Der Löschvorgang selbst ist abhängig von der Art des zu löschenden Modellelementes. Zum Beispiel werden frei stehende Referenzen ohne Rückfrage gelöscht, Originale werden in zwei Stufen gelöscht. Dabei erfolgt zuerst das Entfernen aus dem Diagramm und dann auf Anfrage auch das Löschen aus der Datenbank. Objekte die verfeinert oder verknüpft sind lassen sich nicht ohne erhöhten Aufwand aus dem Modell entfernen. Zuerst müssen die Modellbeziehungen des Elementes aufgelöst werden bis es für den Löschvorgang isoliert ist. Dazu springt in entsprechenden Fällen der Löschassistent an. Dieser gibt dem Anwender die Möglichkeit zuerst die Auswirkungen auf das Gesamtmodell zu beurteilen und kann dann auf Anforderung komplexe Löschvorgänge, zum Beispiel von ganzen Teilmodellen in der Modellhierarchie, durchführen.

3.4.8 Modellelemente kopieren

Um eine Kopie eines Modellelementes zu erzeugen wählen Sie im Menü oder in der Werkzeugleiste den Befehl „Kopieren“. Sie können dieses Objekt jetzt innerhalb des Projektes in ein Diagramm passenden Typs einfügen. Sollte das Diagramm den Objekttyp nicht akzeptieren, erhalten Sie dazu eine Meldung. Rahmenelemente wie zum Beispiel Klassen können mit deren Inhalt kopiert werden. Dabei werden jedoch Modellelemente in der Verfeinerung nach unten nicht berücksichtigt (flache Kopie). Innerhalb eines Diagramms können Sie von einzelnen Objekten durch Drag und Drop bei gleichzeitig gedrückter Steuerungstaste Kopien erzeugen.

Hinweis:

Beachten Sie, dass eine Kopie immer dem kopierten Element entspricht. Das heißt, die Kopie eines Originals ist wieder ein neues Original, die Kopie einer Referenz ist eine neue Referenz auf das ursprüngliche Original.

3.4.9 Modellelemente ausschneiden

Zum Ausschneiden eines Modellelements wählen Sie im Menü oder der Werkzeugleiste *Bearbeiten/Objekt/Ausschneiden*. Sie können dieses Objekt jetzt innerhalb des Projektes in ein Diagramm passenden Typs einfügen. Sollte das Diagramm den Objekttyp nicht akzeptieren, erhalten Sie dazu einen Meldung. Rahmenelemente wie zum Beispiel Klassen können mit deren Inhalt verschoben werden. Dabei werden auch Modellelemente in der Verfeinerung nach unten berücksichtigt.

Hinweis:

Beachten Sie, dass dieser Vorrang starke Auswirkungen auf die Modellstruktur hat. Vergewissern Sie sich über den Objektreport über die möglichen Auswirkungen.

3.4.10 Diagramme kopieren (flache Kopie)

Sie können Kopien von gesamten Diagrammen anlegen und einfügen. Dabei unterscheidet man zwischen einer flachen Kopie bei der alle Elemente der gezeigten Ebene (Diagramm) kopiert werden und einer tiefen Kopie bei der alle Elemente dieser Ebene und der darunter liegenden Ebenen (Teilhierarchie) kopiert werden. Eine flache Kopie erzeugen Sie über die Menübefehle *Bearbeiten/Diagramm/Diagrammkopie anlegen (flache Kopie)* und *Bearbeiten/Diagramm/Diagrammkopie einfügen (flache Kopie).*

Hinweis:

Beachten Sie, dass das Einfügen der Diagrammkopie nur in einem Diagramm des gleichen Typs und innerhalb des gleichen Projektes möglich ist.

3.4.11 Modelle kopieren (tiefe Kopie)

Sie können Kopien von gesamten Teilmodellen anlegen und einfügen. Dabei unterscheidet man zwischen einer flachen Kopie bei der alle Elemente der gezeigten Ebene (Diagramm) kopiert werden und einer tiefen Kopie bei der alle Elemente dieser Ebene und der darunter liegenden Ebenen (Teilhierarchie) kopiert werden. Eine tiefe Kopie erzeugen Sie über die Menübefehle *Bearbeiten/Diagramm/Diagrammkopie anlegen (tiefe Kopie)* und *Bearbeiten/Diagramm/Diagrammkopie einfügen (tiefe Kopie).*

Hinweis:

Beachten Sie, dass eine tiefe Kopie dem Wesen nach ein Export und Import von Teilmodellen darstellt. Das Einfügen eines solchen Imports in ein anderes Projekt kann zu Modellkonflikten führen.

3.4.12 Die Modellhierarchie bearbeiten (JoJo)

Die Modellhierarchie kann durch Einfügen neuer Ebenen oder Auflösen von Modellebenen verändert werden (JoJo). Diese Funktion erreichen Sie am besten über das Haupt- oder Kontextmenü. Um eine gewünschte Anzahl von Modellelementen zu einer neuen Ebene zusammenzufassen, selektieren Sie diese und wählen im Menü *Bearbeiten/Objekt/JoJo/Objekte zusammenfassen*. Es wird ein neues Objekt erzeugt und in das

Diagramm eingefügt. Die ausgewählten Objekte wurden dem neuen Objekt untergeordnet. Bei der Funktion *JoJo/Objekt auflösen* werden die untergeordneten Objekte in das aktuelle Diagramm eingefügt und das ausgewählte Objekt gelöscht.

3.5 Die Ansicht verändern

3.5.1 Fensterinhalt verschieben

Modelle werden oft viel größer als der sichtbare Bereich des Diagrammfensters. Um den sichtbaren Teil eines Diagramms zu verändern (Scroll) haben Sie folgende Möglichkeiten:

- die Scrollleisten des Fensters benutzen
- den Diagrammhintergrund per Drag und Drop schieben
- wenn kein Modellelement ausgewählt ist, mit den Cursortasten scrollen
- Menüpunkt *Ansicht/Fenstermitte* um zum Diagrammursprung zurück zu kehren

Hinweis:
Beachten Sie, dass der Diagrammursprung nicht wie bei einer Textverarbeitung oben links des Arbeitsbereiches liegt sondern wie bei einer MindMap oder vielen Konstruktionsprogramm (CAD) in der Mitte (Nullpunkt, Fenstermitte) und das Modell sich somit in alle Richtungen entwickeln lässt. Vergleichen sie dazu die Position der Scrollbalken. Versuchen Sie auf keinen Fall zu Beginn der Arbeit die Scrollposition oben links herzustellen!

3.5.2 Fensterinhalt vergrößern oder verkleinern

Um einen Überblick über größere Diagramme zu gewinnen, können Sie die Darstellung vergrößern oder verkleinern (Zoom). Eine elegante Möglichkeit das aktuelle Diagramm zu zoomen ist das Scrollrad der Maus. Sie können aber auch über das Kontextmenü oder den Menüpunkt *Ansicht* feste Zoom-Faktoren einstellen. Die Funktion *Ansicht/Zoom/Alles* aus dem Menü oder der Werkzeugleiste passt das Diagramm an die Fenstergröße an. Eine weitere Möglichkeit das Diagramm zu zoomen, erhalten Sie über die Tasten des Numerikfeldes:

- Numerikfeld **+** ... Ansicht vergrößern
- Numerikfeld **-** ... Ansicht verkleinern
- Numerikfeld ***** ... Ansicht alles (einpassen)

3.5.3 Fensterinhalt farbig oder als Kontur darstellen

Temporär kann die farbige Darstellung von Elementen unterbunden werden. Dazu können Sie über das Kontextmenü oder das Menü *Ansicht/Darstellung/Kontur* zwischen farbiger oder Konturdarstellung umschalten.

3.5.4 Die Schriftart ändern

Die in Diagrammen verwendete Schriftart lässt sich über die Werkzeugleiste oder den Menüpunkt *Einstellungen/Schriftart…* verändern. Dabei sollte die Schriftgröße 10 bis 12 eingehalten werden.

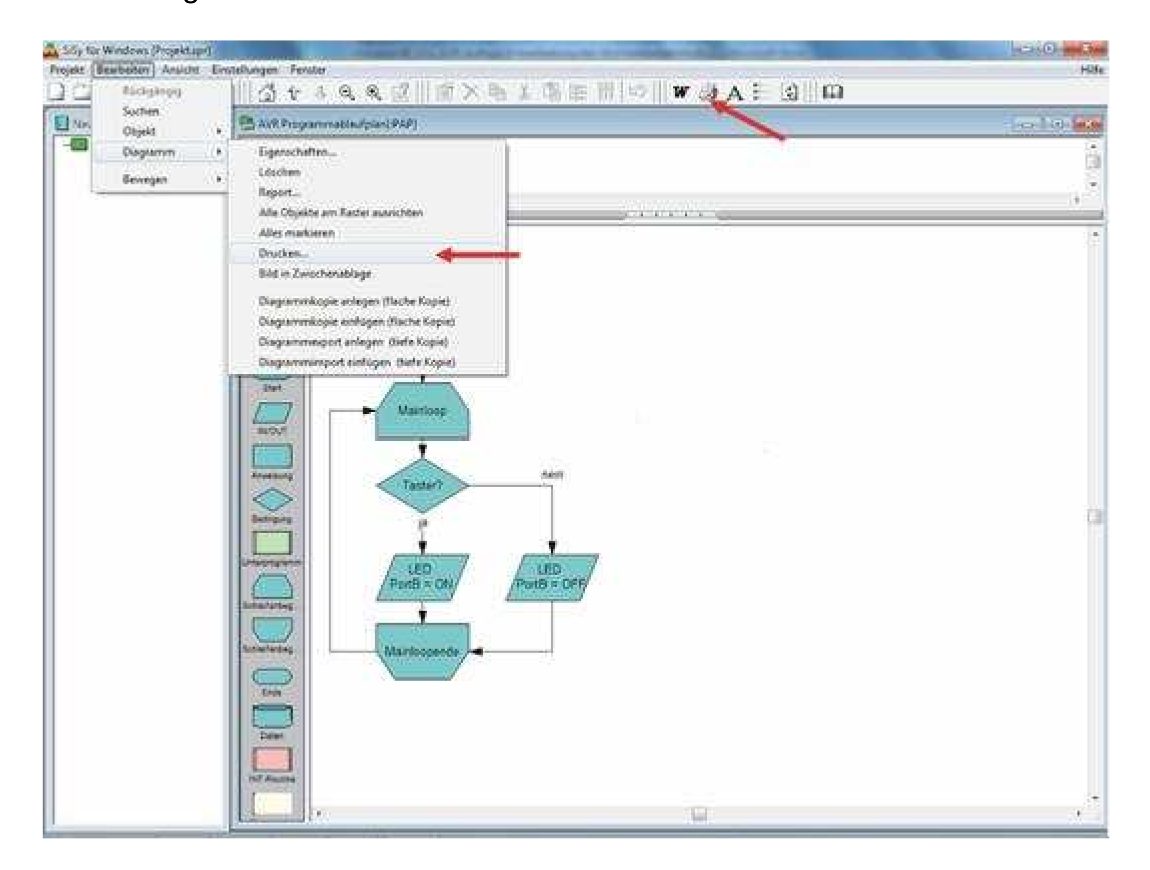
3.6 Druckfunktionen in SiSy

Sie haben in SiSy verschieden Möglichkeiten Projektinhalte wie Grafiken, Übersichten, Quellcodes oder ganze Projektdokumentationen zu drucken. Dabei ist zu berücksichtigen, dass in SiSy bestimmte Informationen wie zum Beispiel die Darstellung eines Programmablaufplanes sichtbare Elemente eines Diagramms sind und andere Teile wie zum Beispiel der Quellcode eines Elementes nur über Dialoge oder bei Selektierung des Elementes sichtbar sind. Je nachdem, welchen Inhalt Sie dokumentieren wollen, richtet sich die Auswahl der betreffenden Druckfunktion.

3.6.1 Diagramme drucken

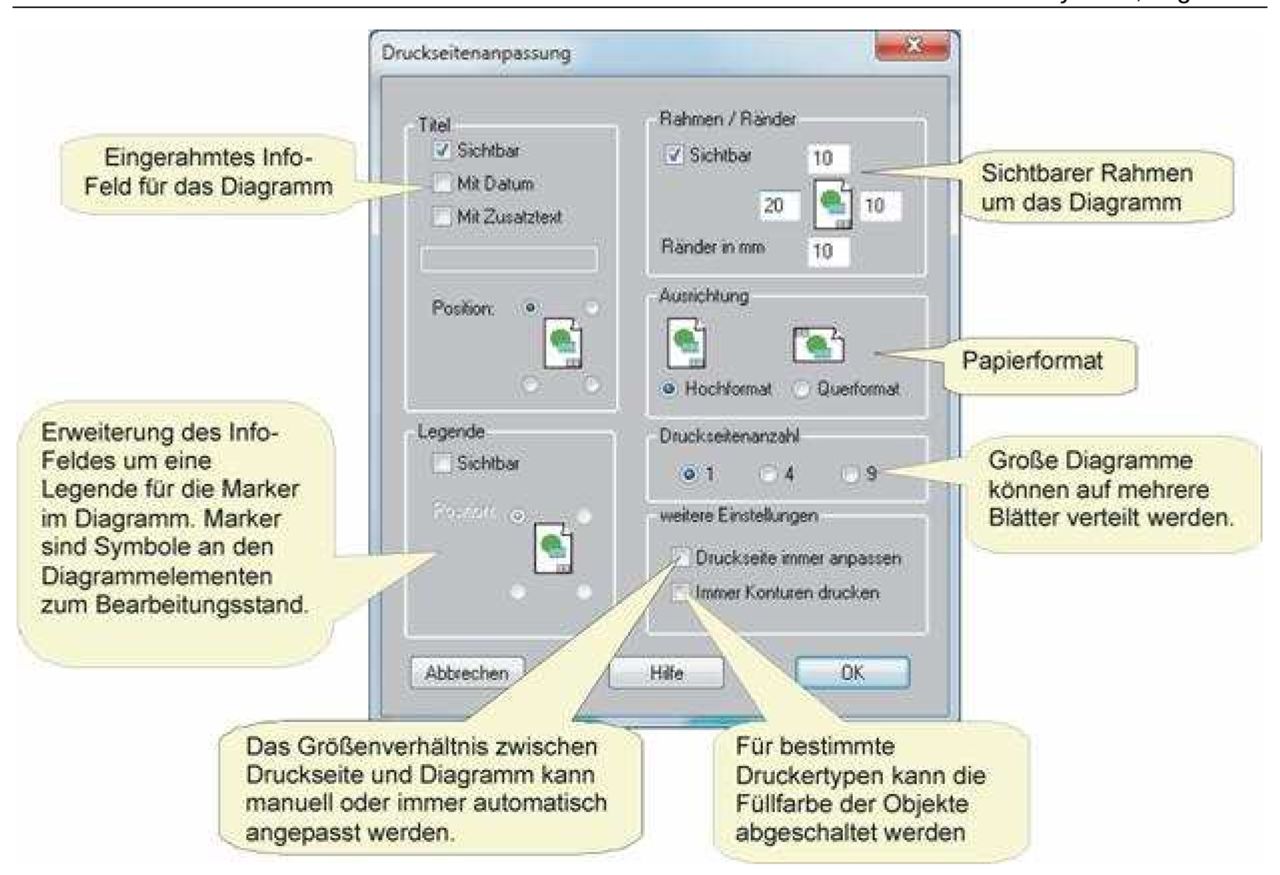
Wenn Sie ein einzelnes Diagramm, also den sichtbaren Inhalt eines Diagrammfensters drucken wollen, gehen Sie wie folgt vor:

- ggf. Projekt öffnen
- das gewünschte Diagramm öffnen
- die Menüfolge Bearbeiten/Diagramm/Drucken… wählen oder das Druckersymbol in der Werkzeugleiste aktivieren



- Sie erhalten den Dialog zum Einrichten der Druckseite, wählen Sie die gewünschten Optionen aus!



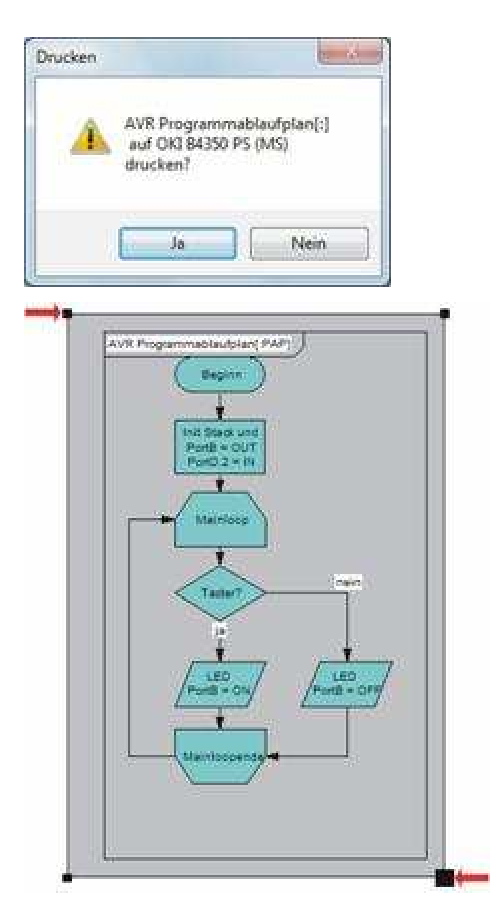


- Zum Drucken wählen Sie die Schaltfläche „OK“. Der Ausdruck erfolgt auf dem Standarddrucker des Systems

- Im Hintergrund ist die Druckvorschau zu sehen. Der Druckvorgang kann hier abgebrochen werden, um die Einstellungen zu überarbeiten. Dabei kann die relative Position und das Größenverhältnis der Druckseite zum Diagramm verändert werden. Der Dialog zum Verändern der Einstellungen lässt sich per Doppelklick auf den Selektierungsmarken der Druckseite öffnen.

- Die Druckseitenansicht lässt sich über die Menüfolge Ansicht/Druckseite ein- und ausblenden

- Der Druckvorgang kann über das Druckersymbol in der Werkzeugleiste jederzeit gestartet werden.



3.6.2 Grafiken und Inhalte drucken (QuickDok)

Viele Projektinformationen sind kein sichtbarer Teil von erstellten Diagrammen. Diese wurden über Dialoge und Masken eingegeben und stehen als Attribute in der Projektdatenbank zur Verfügung. Um diese Informationen auszudrucken, stellt SiSy für jeden Diagrammtyp eine Reportfunktion zur Verfügung, mit der die wesentlichen Informationen, Inhalte und Attribute des Diagramms und der Objekte in einem Diagramm, als Word-Dokument generiert werden.

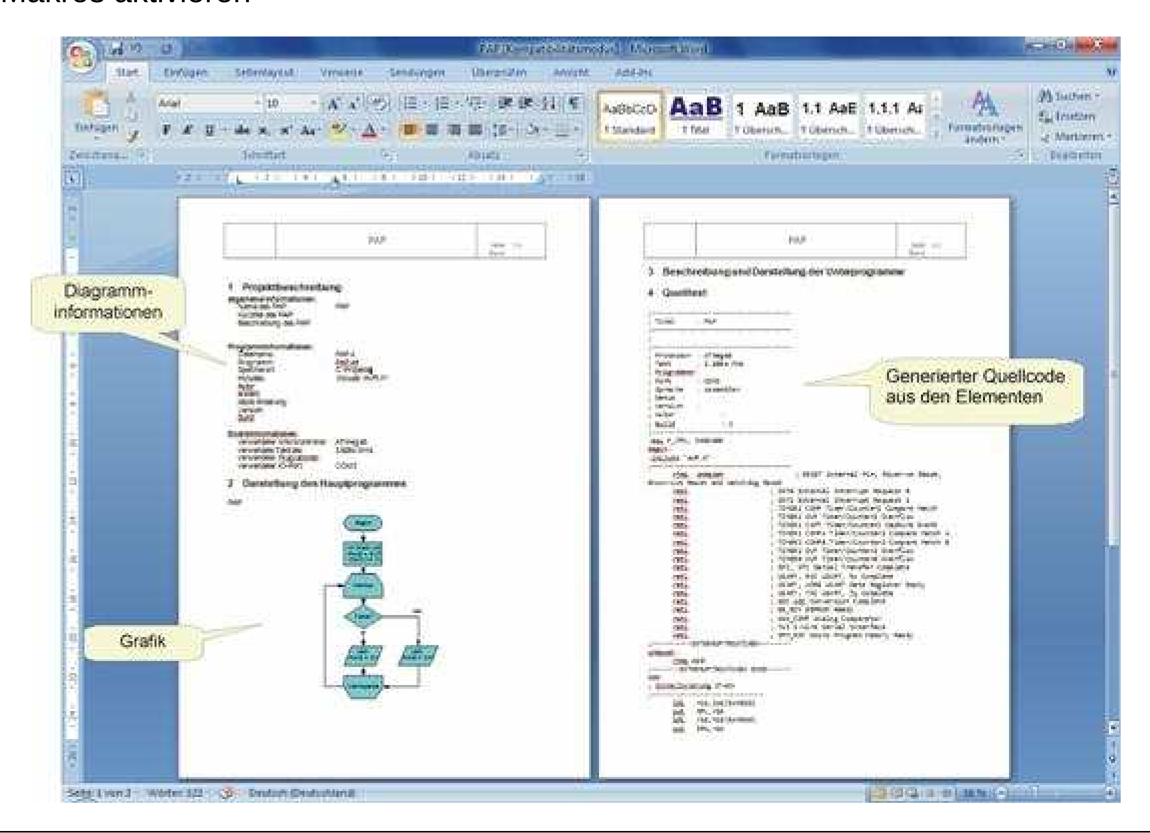
Für das Generieren des Word-Dokumentes muss auf dem PC MS Word installiert sein. Unterstützt wird die Version Word 2007.
Die Dokumentengenerierung basiert auf Makros.
Vorgehensweise in Word:

- Wählen Sie die Schaltfläche „Office“
- Menüfolge: Word-Optionen / Vertrauensstellungscenter / Einstellungen für das Vertrauensstellungscenter / Einstellungen für Makros
- Aktivieren Sie unter „Einstellungen für Makros“ den Auswahlpunkt „Alle Makros aktivieren“
- Setzen Sie im Unterpunkt „Entwicklermakroeinstellungen“ den Haken bei „Zugriff auf das VBA-Projektobjektmodell vertrauen“

Die Reportfunktion zum Generieren des Word-Dokumentes aktivieren Sie über das Symbol „QuickDok“ in der Werkzeugleiste.

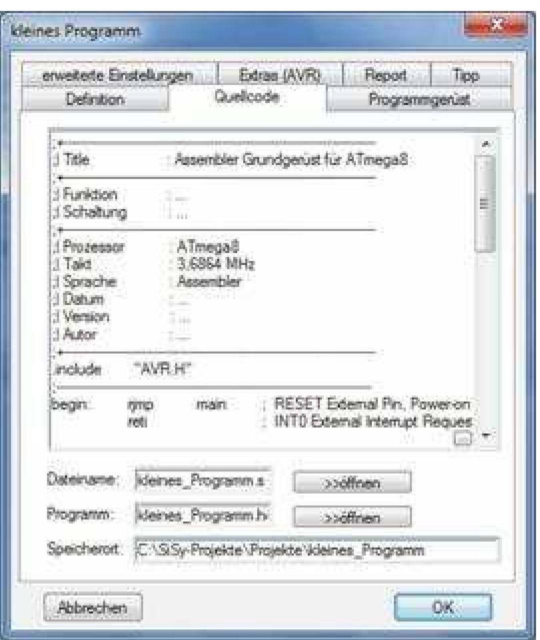
- Symbol W (QuickDok) in der Werkzeugleiste wählen
- Aus der Liste der angebotenen Reportfunktionen auswählen
- Dateinamen für den Report vergeben
- Makros aktivieren





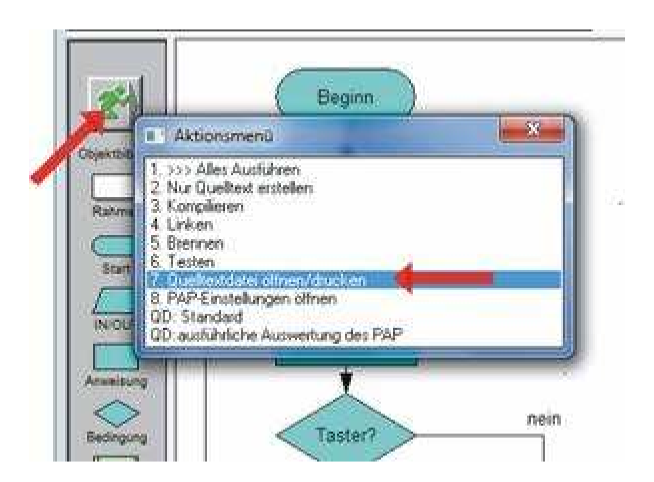
3.6.3 Nur Quellcodes drucken

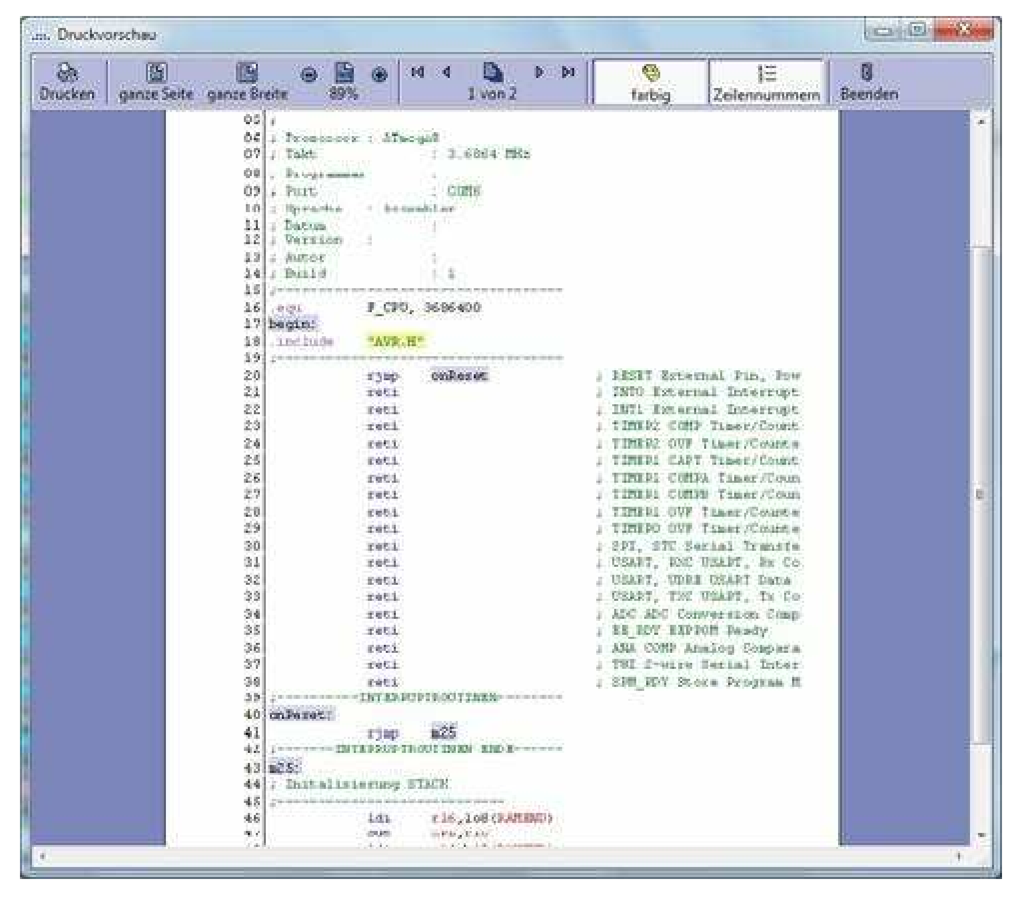
Für das Ausdrucken von Quellcodes bietet SiSy einen speziellen Quellcode-Druckassistenten. Damit ist es möglich Quellcodes formatiert, in Syntaxfarben und mit Zeilennummern auszudrucken.



- kleines Programm, Quellcode drucken
  - o kleines Programm selektieren
  - o rechte Maustaste „Definieren…"
  - o Dialogseite „Quellcode"
  - o Schaltfläche „>>öffnen"

- Programmablaufplan, Quellcode drucken
  - o PAP öffnen
  - o grünes Männchen aktivieren
  - o „Quellcodedatei öffnen/drucken"

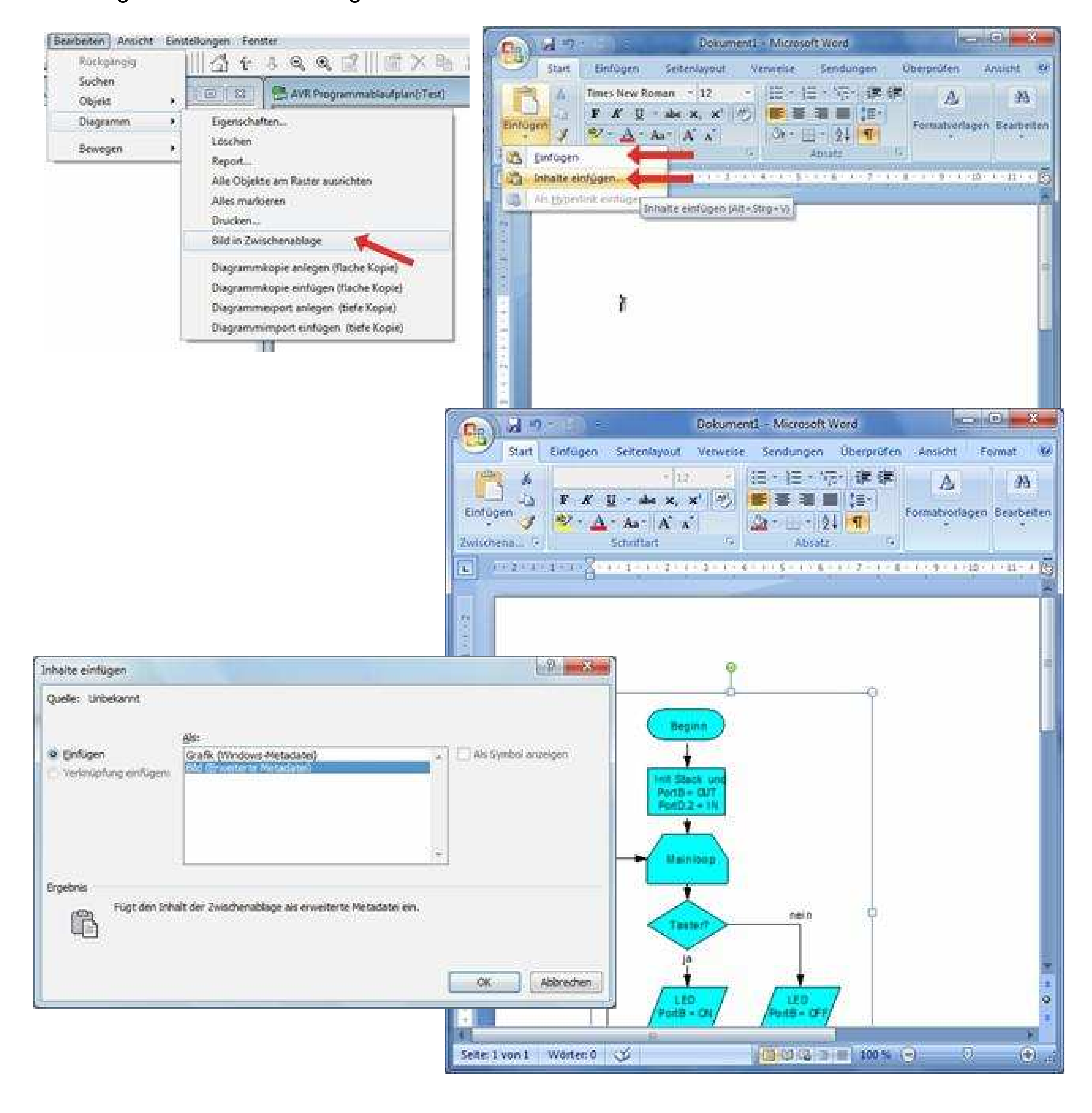




3.6.4 Nutzen der Zwischenablage

Oft ist es erforderlich in Projektdokumentationen die Diagramme als Bilder einzufügen. In SiSy werden die Diagramme nicht als Bilder gespeichert sondern zur Laufzeit aus den Modellinformationen generiert. Um die Bilder der Diagramme weiter zu verwenden, steht dem Anwender die Funktion „Bild in Zwischenablage" zur Verfügung. Dabei erstellt SiSy eine skalierbare Vektorgrafik (WMF) und legt diese in die Zwischenablage (Copy). Die Grafik kann nun von anderen Anwendungen über den Befehl „Einfügen" (Paste) beliebig weiter verwendet werden.

- gewünschtes Diagramm öffnen
- Menüfolge *Bearbeiten/Diagramm/Bild in Zwischenablage* wählen
- Zielanwendung, zum Beispiel Word öffnen
- Menüfolge *Start/Einfügen* oder *Start/Einfügen/Inhalte einfügen* wählen
- Gegebenenfalls einzufügendes Grafikformat wählen



3.7 SiSy Add-Ons verwalten

3.7.1 Einführung

Das Modellierungswekzeug SiSy besteht aus seinen Kernkomponenten:

- Anwendungssystem SiSy mit
  - Rahmenanwendung
  - Assistent
  - Diagrammfenster
- Laufzeitbibliotheken
  - Datenbanktreiber (Repository)
  - Grafikbibliotheken
  - Metamodell-Engine
  - SiSy BASIC Interpreter
- Allgemeine Hilfe (dieses Benutzerhandbuch)

… und installierten Add-Ons.

Die Add-On's enthalten zusätzliche Komponenten, SiSy-BASIC-Skripte, erweiterte Hilfen und spezielle Informationen für die Modellierung (Metamodell). Dabei können Add-On's wiederum aus Add-On's thematisch zusammengestellt werden. Diese zusätzlichen Bausteine werden während der Laufzeit vor allem durch die Metamodell-Engine verarbeitet. Dadurch können beliebige Modellierungsaufgaben durch SiSy verarbeitet werden. Es ist nur das entsprechende Add-On erforderlich. Für die aktuelle Version von SiSy sind zum Beispiel folgende Add-On Zusammenstellungen verfügbar:

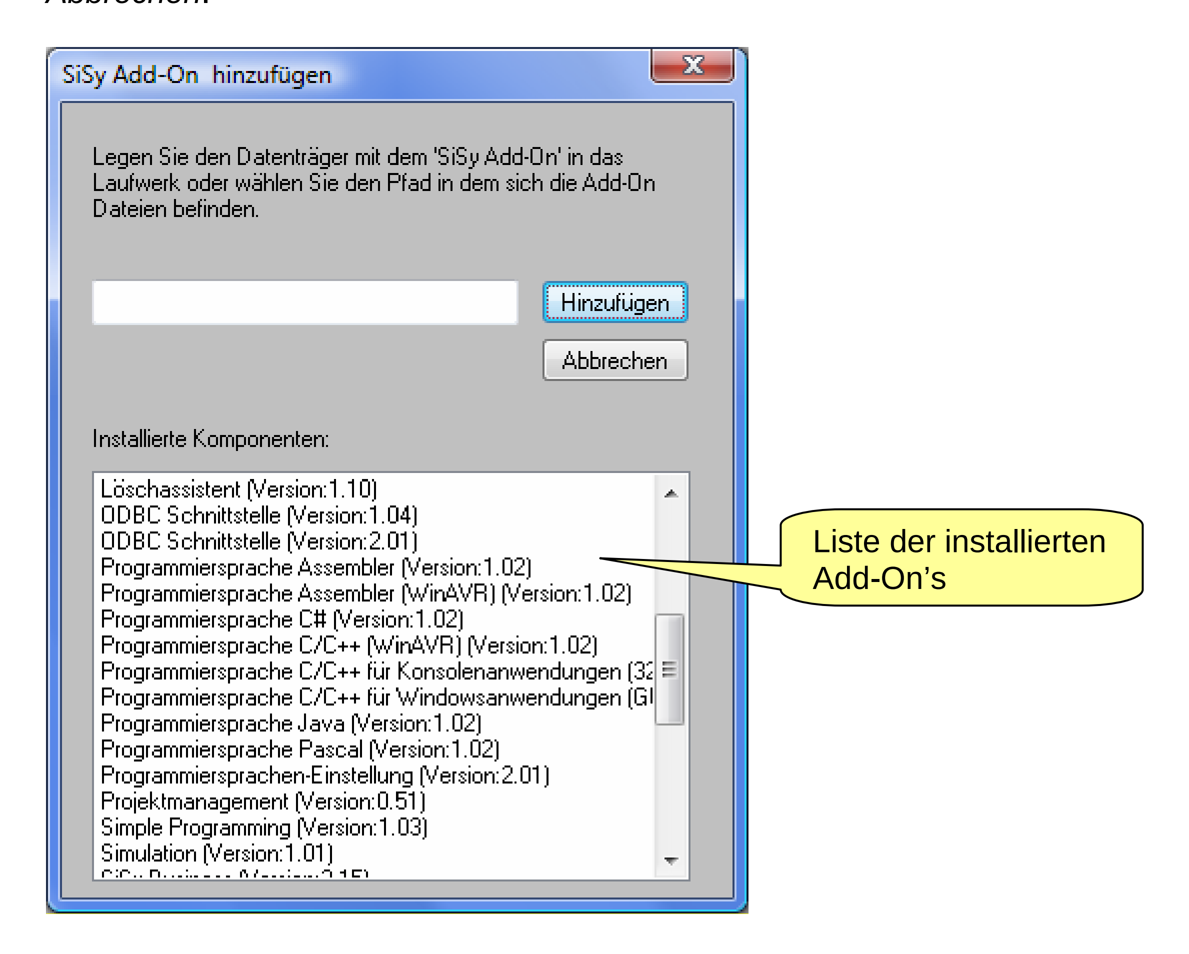
- **AVR**, Programmierung von AVR Mikrocontrollern
  - einfache Programmierung mit dem Zeileneditor in Assembler und C
  - grafische Programmierung mit dem Programmablaufplan in Assembler, dem Struktogramm in C, dem UML-Klassendiagramm in C++ und dem UML-Statechart in C++
  - Codegenerierung mit dem myAVR Code-Wizard für Assembler und C
  - myAVR Werkzeuge für Test und Kommunikation mit AVR Controllern
- **SPRG**, einfache Programmierung für Konsolen- und GUI-Anwendungen
  - C / C++, JAVA, C#, PASCAL, Assembler
  - Integrierter Debugger
- **UML**, objektorientierte Systementwicklung für Konsolen- und GUI-Anwendungen
  - Typische UML-Diagramme z.B.: Klassendiagramm, Anwendungsfalldiagramm, Aktivitätsdiagramm, Zustandsdiagramm, Sequenzdiagramm, u.a.
  - Codegenerierung für C++, C#, JAVA
  - Integrierter Debugger mit Modellnachführung im Schrittbetrieb
- **SVL**, Smart Visual Library
  - Einfache bausteinartige Entwicklung von Windows-Anwendungen (RAD Rapid Application Development) im UML Klassendiagramm
- **SysML**, System Modeling Language
  - Fachübergreifende Systementwicklung mit dem OMG Sprachstandard
- **SA/SD**, Strukturierte Techniken zur Systementwicklung
  - Round Trip Engineering mit den klassischen Analyse und Entwurfswerkzeugen bis zur Codegenerierung, Strukturierte Analyse, Entity Relationship Diagramm mit SQL-Generierung und Struktogramme mit Codegenerierung
- **DOKGEN**, Dokumentengenerierung
  - Generieren von Dokumentationen direkt aus den Modellen heraus.
- **GPM / QM**, Geschäftsprozessmodellierung und Qualitätsmanagement
  - Modellierung von integrierten Managementsystemen

Aus den verfügbaren Add-Ons werden entsprechend der unterschiedlichen Einsatzgebiete des Modellierungswerkzeuges SiSy spezielle Ausgaben zusammengestellt. Die jeweiligen Ausgaben können jederzeit durch weitere Add-Ons ergänzt werden. Es sind unter anderem folgende SiSy-Ausgaben verfügbar (Stand Mai 2011):

	Ausgabe	**Preis**
-	SiSy Professional *umfasst alle verfügbaren Add-Ons*	2800,- €
-	SiSy Business *umfasst Add-Ons für Prozessmodellierung, Qualitäts- und Projektmanagement*	2300,- €
-	SiSy Developer *umfasst umfangreiche Add-Ons zur Systementwicklung*	499,- €
-	SiSy AVR ++ *umfasst die Add-Ons AVR, ausgewählte Teile der UML, SVL und SysML*	199,- €
-	SiSy AVR *beinhaltet nur das Add-On AVR*	99,- €

3.7.2 Add-Ons anzeigen

Um die Liste der installierten Add-On's anzuzeigen nutzen Sie den Menüpunkt *Einstellungen/Add-Ons hinzufügen*. Um den Dialog zu verlassen wählen Sie die Schaltfläche *Abbrechen*.

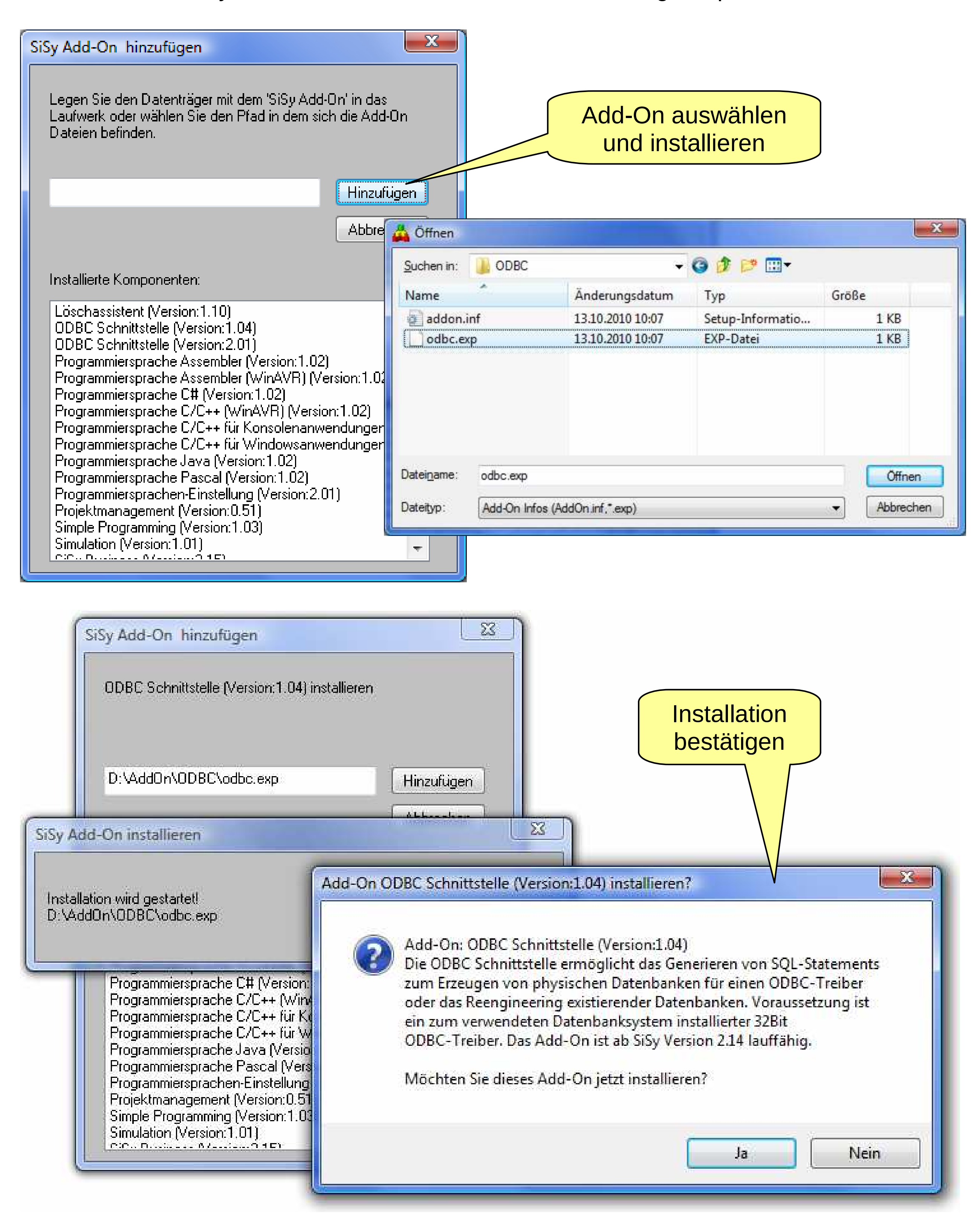


<u>*Hinweis:*</u>

Beachten Sie, dass Änderungen an den Add-On's in jedem Fall Auswirkungen auf die Funktionalität von SiSy und abhängiger Add-On's haben.

3.7.3 Add-Ons hinzufügen

Für das Hinzufügen von Add-On's nutzen Sie den Menüpunkt *Einstellungen/Add-Ons hinzufügen*. Wählen Sie im Add-On Dialog die Schaltfläche *Hinzufügen* um ein Add-On zu installieren. SiSy Add-On's besitzen die Dateierweiterung *.exp.



<u>*Hinweis:*</u>

Für die Installation von Add-On's sollten Sie Administratorrechte besitzen. Beachten Sie, dass Änderungen an den Add-On's in jedem Fall Auswirkungen auf die Funktionalität von SiSy und abhängiger Add-On's haben.

3.8 Zusätzliche Werkzeuge

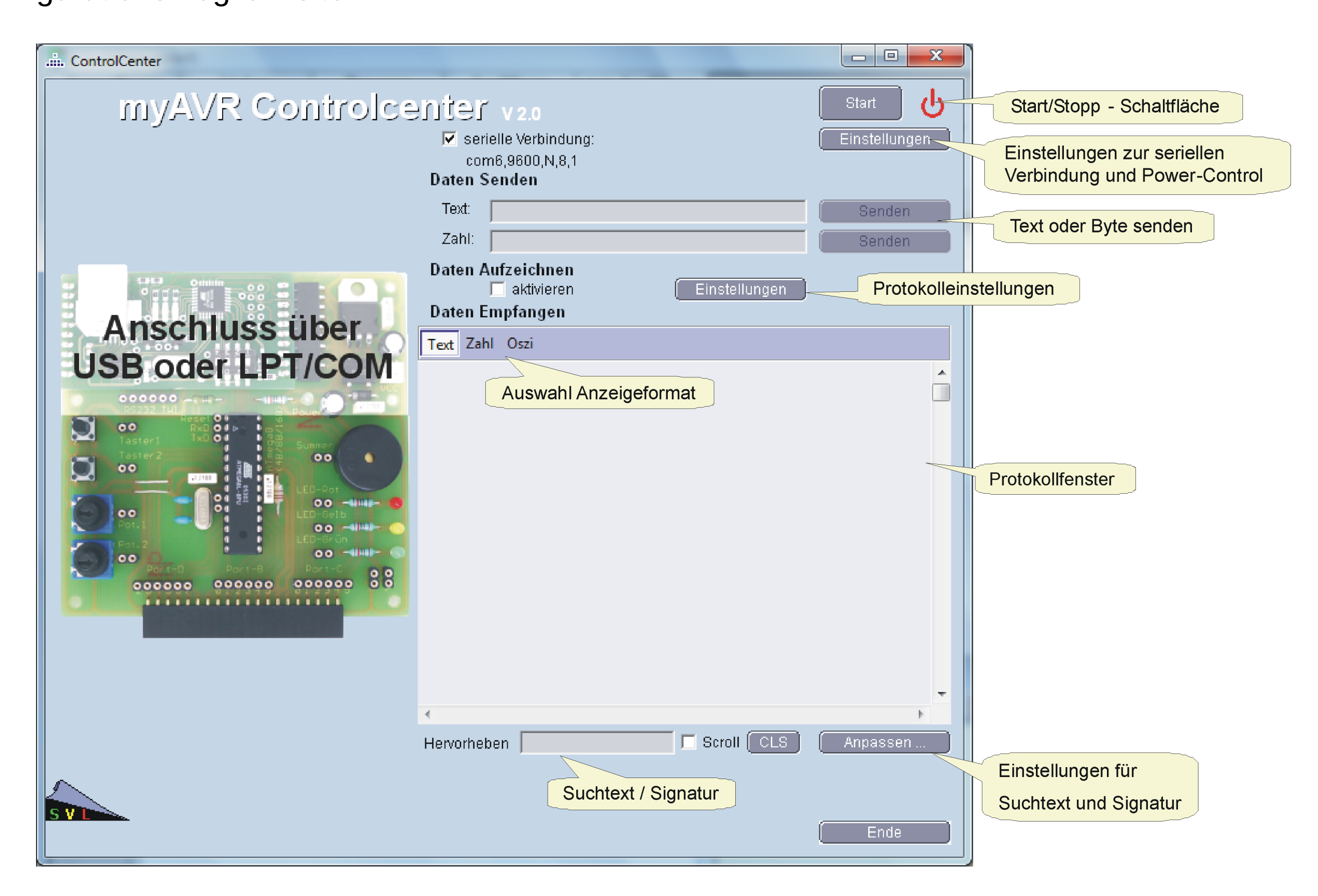
3.8.1 Einführung

Bestimmte Add-On's installieren als zusätzliche Komponenten externe Programme. Diese sind nach der Installation über den Menüpunkt *Werkzeuge* erreichbar.

3.8.2 myAVR Werkzeuge

Das myAVR ControlCenter

Das myAVR Controlcenter ist ein universelles Terminalprogramm zur Kommunikation mit dem myAVR Board und anderen Mikrocontrollerapplikationen, die über eine serielle Schnittstelle (UART) oder USB Anbindung mit virtuellem COM-Port zum PC verfügen. Es kann für Test- und Debug-Meldungen sowie Visualisierung und Protokollierung von Messdaten genutzt werden. Dazu bietet das myAVR Controlcenter umfangreiche Konfigurationsmöglichkeiten.



Sie finden im Kapitel „Das myAVR Controlcenter" ausführliche Informationen für dessen Handhabung. Das Controlcenter kann einzeln über das Menü Werkzeuge oder im Zusammenhang mit einer Mikrocontrolleranwendung aus dem jeweiligen Diagramm für die Entwicklung des Programms (z.B. Klassendiagramm, Programmablaufplan, usw.) aufgerufen werden.

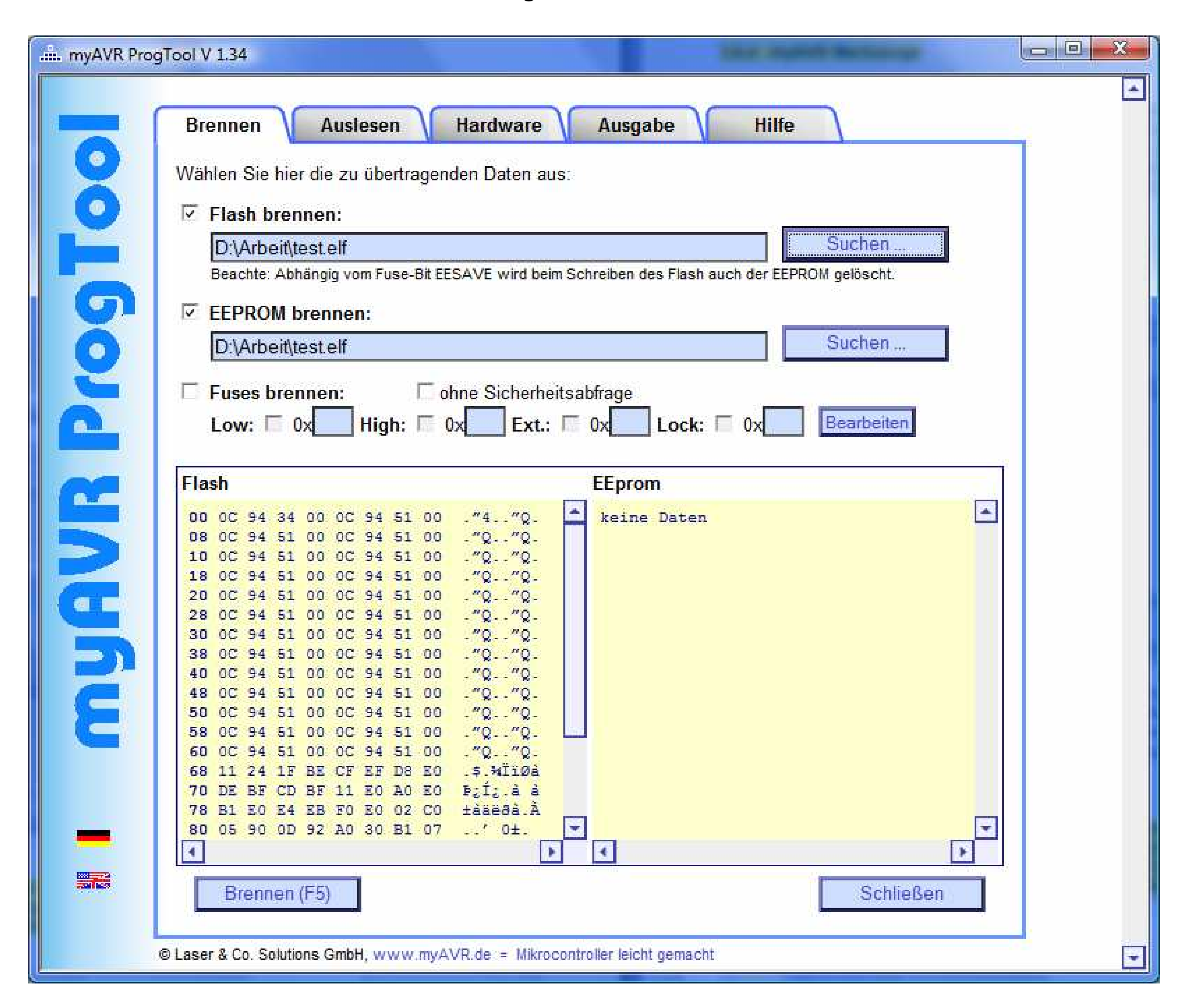
<u>*Hinweis:*</u>
Das myAVR Controlcenter verfügt über spezielle Fähigkeiten die myAVR Programmer vom Typ MK2 und MK3 anzusteuern. Informieren Sie sich über die Möglichkeiten der Spannungsversorgung und der Kommunikation dieser Programmer in den jeweiligen technischen Beschreibungen. Diese Dokumente stehen Ihnen im Downloadbereich von www.myAVR.de zur Verfügung.

Das myAVR ProgTool

Das myAVR ProgTool ist ein Werkzeug zum Programmieren von AVR-Mikrocontrollern. Sie können den Programmspeicher, den EEPROM und die Fuse-/Lock-Bits der unterstützten AVR-Mikrocontroller komfortabel programmieren.

Brennen:

Der Inhalt der ausgewählten Dateien wird im unteren Bereich des Fensters angezeigt. Die Inhalte der Eingabefelder bleiben nach dem Schließen erhalten und sind nach erneutem Öffnen des Tools wieder verfügbar.



Signalfarben:

rötlich gefülltes Datei-Eingabefeld: Datei existiert nicht
bläulich gefülltes Datei-Eingabefeld: Datei existiert
graue Schrift: Element inaktiv

Flash brennen:

Aktivieren Sie zunächst die Auswahl "Flash brennen". Wählen Sie anschließend per Schaltfläche "Suchen..." die gewünschte Datei (*.elf, *.hex, *.raw, *.bin) aus. Der Dateiname kann ebenso per Hand in das Eingabefeld eingegeben werden.
Bei Auswahl einer *.elf-Datei, wie sie zum Beispiel mit WinAVR erstellt wird, erfolgt eine automatische Konvertierung in das Intel-HEX-Format. Sind in der ELF-Datei die Sektionen ".eeprom", ".fuses" bzw. ".lock" enthalten, werden diese extrahiert und als Werte für EEPROM und die Fuses verwendet.

EEPROM brennen

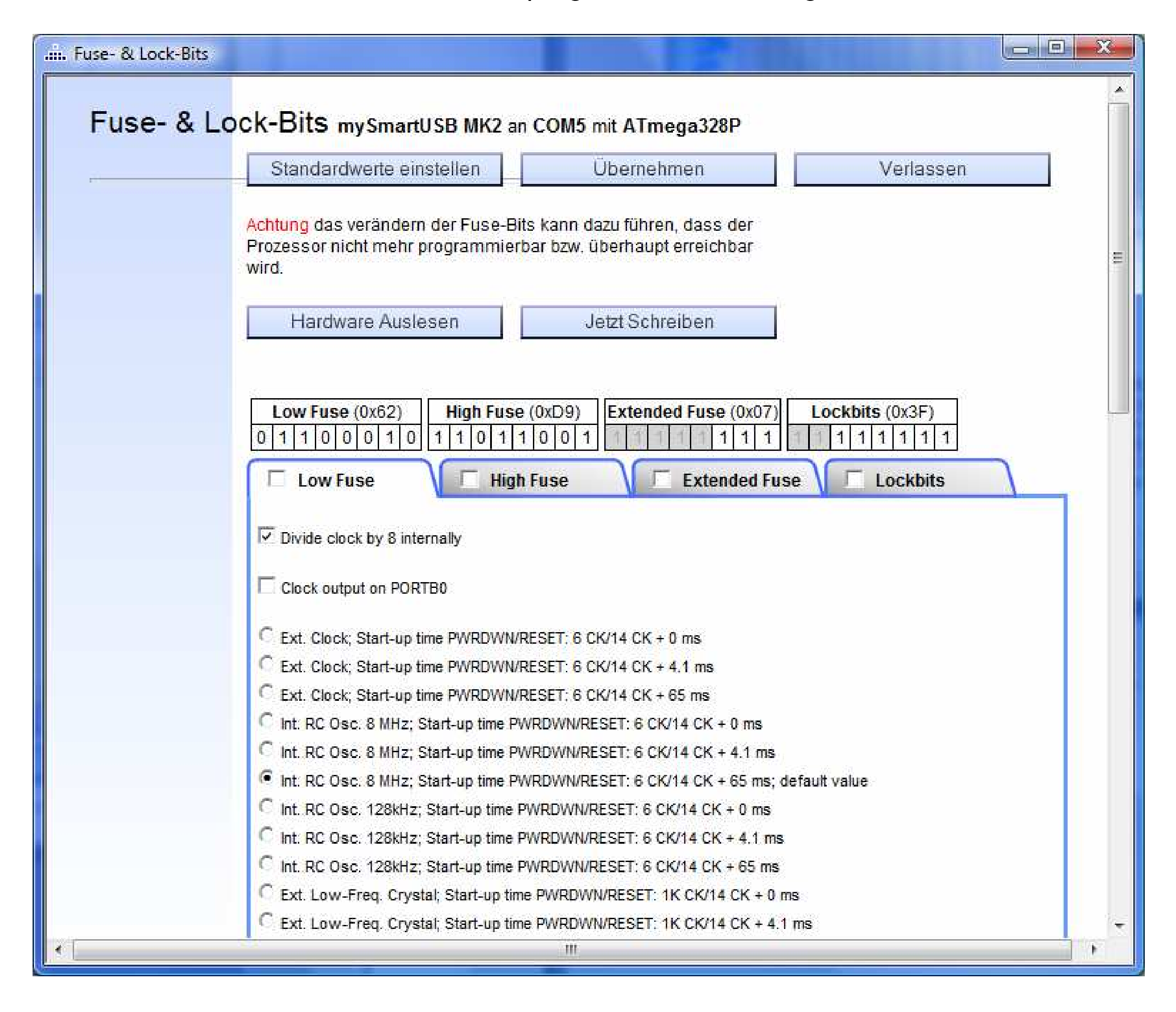
Aktivieren Sie zunächst die Auswahl "EEPROM brennen". Wählen Sie anschließend per Schaltfläche "Suchen..." die gewünschte Datei (*.eep, *.elf, *hex, *.raw, *.bin, *.txt) aus. Der Dateiname kann ebenso per Hand in das Eingabefeld eingegeben werden.

Fuses brennen:

Sie ereichen die Fuse- und Lock-Bit Konfiguration über "Bearbeiten" auf der Seite "Brennen". Aktivieren Sie zunächst die Auswahl "Fuses brennen" sowie die gewünschten Fuses (Low-, High- und Extended Fuse sowie Lock-Bits). Tragen Sie die hexadezimalen Werte entsprechend der Datenblätter des zu programmierenden Mikrocontrollers ein. Sie können die Werte auch über die Schaltfläche "Bearbeiten" einstellen.

Sie können die Fuse- und Lock-Bits in einer Listenansicht nach Ihren Bedürfnissen konfigurieren. Dazu muss vorher im Reiter "Hardware" der korrekte Controller ausgewählt sein. Wollen Sie die aktuellen Werte auslesen oder verändern, müssen Sie auch den richtigen Programmer einstellen und anschließen.

Beachten Sie, dass bei fehlerhaften Einstellungen der Fuse- und Lock-Bits der Mikrocontroller unter Umständen nicht mehr programmierbar oder gar erreichbar sein kann.



Auslesen eines Controllers:

Zum Auslesen muss die richtige Hardware (Controller und Programmer) eingestellt sein. Die aktuellen Einstellungen sehen Sie im Kopfteil der Seite.

Lesen Sie die Daten des Flash- oder EEPROM-Speichers durch Betätigen der jeweiligen Schaltfläche aus. Dieser Vorgang kann je nach Speichergröße länger dauern. Es wird der komplette Speicher ausgelesen, haben Sie also etwas Geduld.
Nach erfolgreichem Auslesen werden die gelesenen Daten dargestellt. Nicht belegte Speicherbereiche am Ende der Daten werden vor der Darstellung entfernt.

Sie können die ausgelesenen Daten in eine Datei speichern. Die Speicherung erfolgt im Intel-HEX-Format.

Hardware einstellen:
Wählen Sie vor dem Brennen immer die korrekte Programmierhardware und den zu programmierenden Controllertyp aus. Die Einstellungen werden beim Schließen gespeichert und sind nach erneutem Öffnen wieder verfügbar. Mit "Test" bzw. "?" kann der aktuelle Controller und der verwendete Port ermittelt werden.

myAVR ProgTool V 1.34

myAVR ProgTool

Brennen | Auslesen | Hardware | Ausgabe | Hilfe

Stellen Sie hier Programmertyp, ggf. Port und Controllertyp ein.

Programmer:

- mySmartUSB MK3 / myAVR Board MK3 — Anschluss: COM8 ?
- mySmartUSB light — Anschluss: COM3 ?
- mySmartUSB MK2 / myAVR Board MK2 USB / myMultiProg MK2 USB — Anschluss: COM5 ?
- AVR ISP mk-II — Anschluss: usb:48:74 ?
- myAVR Board MK1 LPT / myMultiProg MK1 (LPT) — Anschluss: LPT1 ?
- AVR Dragon — Anschluss: usb:3e:65 ?
- myAVR Bootloader - mySmartControl - myAvrStamp — Anschluss: COM3 ?
- Sonstiges: avrisp - Atmel AVR ISP — Anschluss: COM8

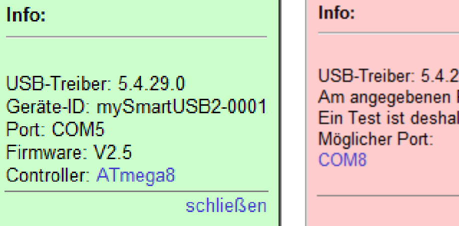
Controller: ATmega328P

Hilfe bei Verbindungsproblemen

Schließen

© Laser & Co. Solutions GmbH, www.myAVR.de = Mikrocontroller leicht gemacht

Signalfarben:
gelb: Test läuft
grün: Test war erfolgreich
rot: Test ist fehlgeschlagen



Info:

USB-Treiber: 5.4.29.0
Am angegebenen Port ist zur Zeit kein Gerät angesteckt.
Ein Test ist deshalb nicht möglich.
Möglicher Port:
COM8

schließen

Besonderheiten:

- mySmartUSB, myAVR Board, myMultiProg : → MK2, MK3, light / mySmartControl MK2: → 8K, 16K, 32K / mySmartControl MK3
  - USB-Treiber muss installiert sein (CP210X)
  - mySmartUSB muss angeschlossen sein
  - virtueller COM-Port (COMx) muss aktiv sein
  - Port-Einstellungen sind nicht notwendig, da der Programmer automatisch gefunden wird
  - Ziel-Hardware wird vom mySmartUSB mit Spannung versorgt
  - Über "Test" kann der verwendete Controller ermittelt werden

- myAVR Board MK1, myMultiProg MK1:
  - LPT-Kabel muss angeschlossen sein
  - Board muss angeschlossen sein
  - evtl. externe Spannungsversorgung anschließen
  - der korrekte LPT-Port muss im Feld "Anschluss:" eingetragen sein
  - Anschluss und Controller können per "?" ermittelt und überprüft werden

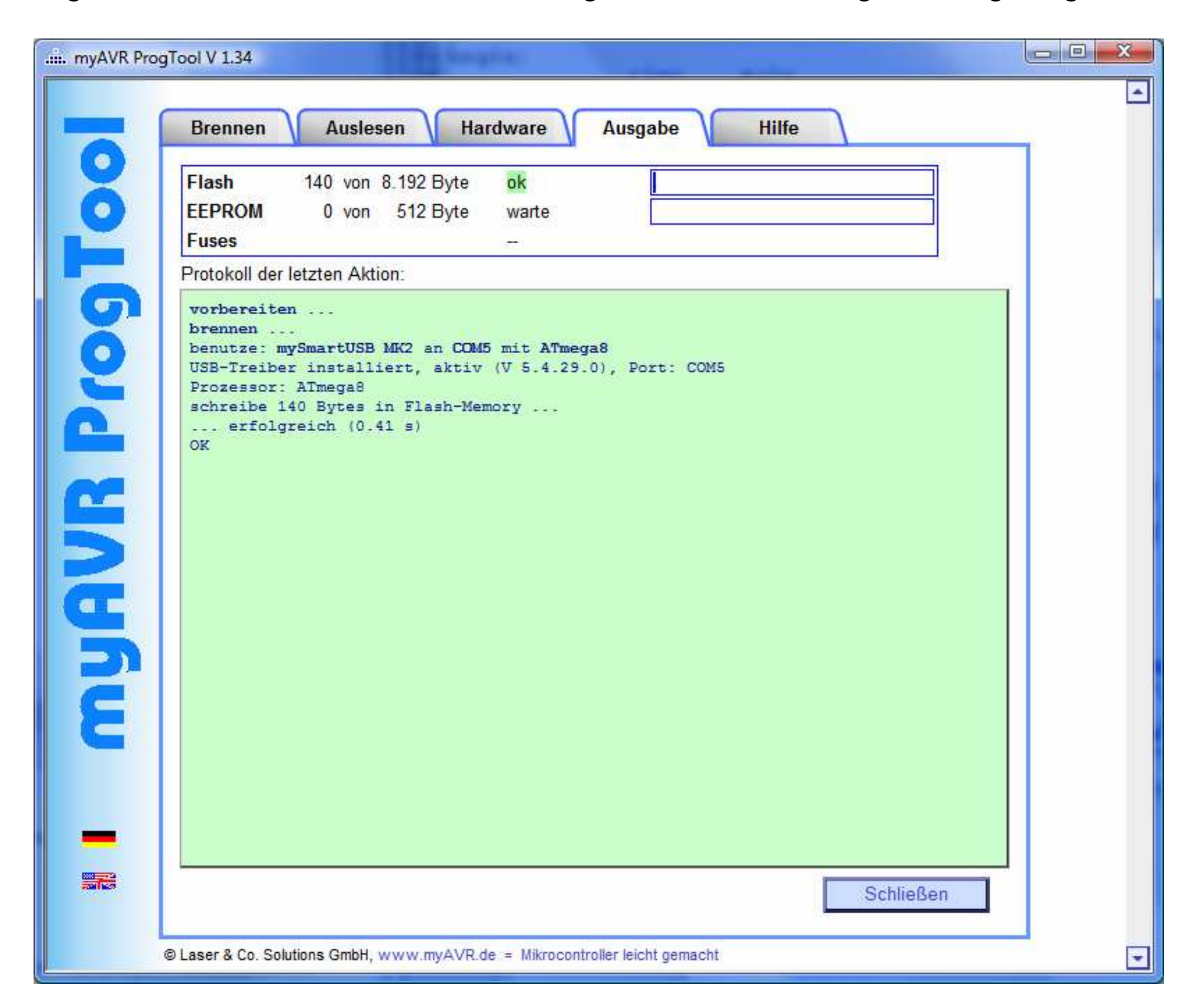
- JTAG ICE mk-II:
  - LIB-USB-Treiber muss installiert sein
  - JTAG ICE mk-II muss angeschlossen sein
  - Spannungsversorgung der Ziel-Hardware muss eingeschaltet sein
  - korrekte USB-Port-Nummer muss im Feld "Anschluss:" eingetragen sein
  - Anschluss und Controller können per "?" ermittelt und überprüft werden

- STK 500:
  - Board muss an einem COM-Port angeschlossen sein
  - externe Spannungsversorgung muss angeschlossen sein
  - Board muss eingeschaltet sein (Power)
  - ISP-Pin muss verbunden sein
  - korrekter COM-Port muss im Feld "Anschluss:" eingetragen sein
  - Anschluss wird nicht automatisch ermittelt

- AVR ISP mk-II:
  - LIB-USB-Treiber muss installiert sein
  - AVR ISP mk-II muss angeschlossen sein
  - Spannungsversorgung der Ziel-Hardware muss eingeschaltet sein
  - korrekte USB-Port-Nummer muss im Feld "Anschluss:" eingetragen sein
  - Anschluss und Controller können per "?" ermittelt und überprüft werden

- AVR Dragon:
  - LIB-USB-Treiber muss installiert sein
  - AVR Dragon muss angeschlossen sein
  - Spannungsversorgung der Ziel-Hardware muss eingeschaltet sein
  - korrekte USB-Port-Nummer muss im Feld "Anschluss:" eingetragen sein
  - Anschluss und Controller können per "?" ermittelt und überprüft werden

Ausgabe:
Die Aktionen werden im Ausgabefenster protokolliert. Neben anderen Informationen werden Anzahl der übertragenen Bytes und Dauer der Aktion angezeigt. Zustand und Ergebnis der Aktionen werden durch Signalfarben und Ausgaben angezeigt.



Signalfarben:
hellgrauer Hintergrund: keine Aktion, wartend
hellgelber Hintergrund: Aktion wird momentan ausgeführt
hellroter Hintergrund: Aktion wurde nicht erfolgreich ausgeführt, Fehler
hellgrüner Hintergrund: Aktion wurde erfolgreich beendet

Problembehandlung:

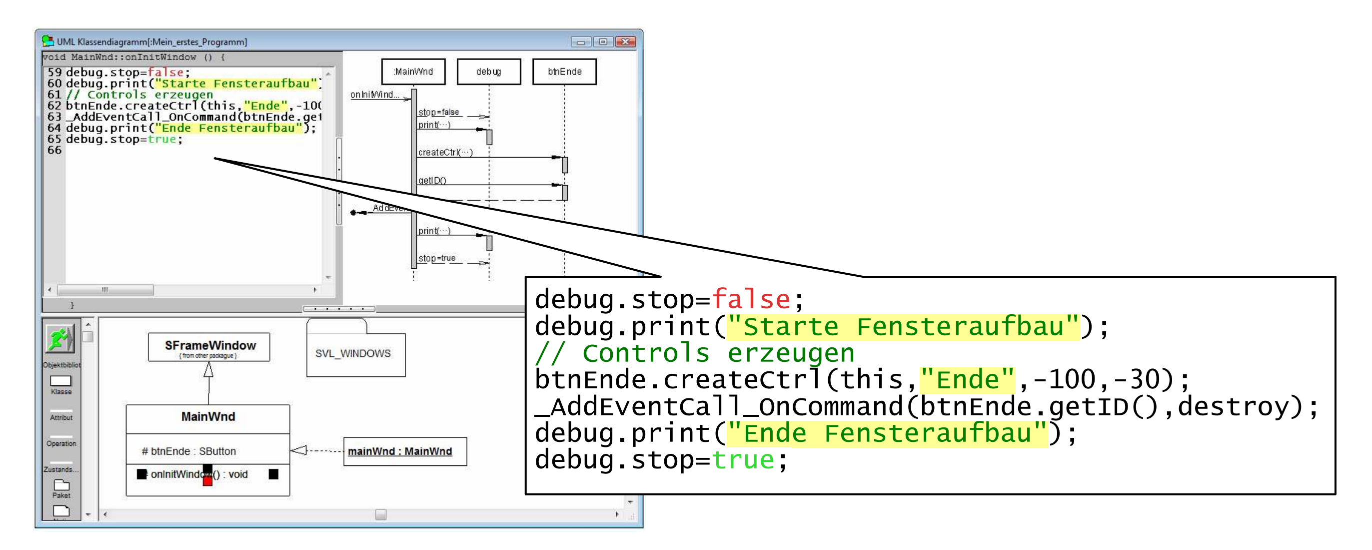
- Allgemein: Überprüfen Sie, ob alle Kabel richtig angeschlossen sind und überprüfen Sie, ob die Stomversorgung der Zielplattform ausreichend ist.
- Bei USB-Verbindungen trennen Sie diese kurz (ca. 20s) um eine Reinitialisierung des USB-Treibers durchzuführen.
- mySmartUSB funktioniert nicht:
  - USB-Treiber (CP210x) nicht installiert -> Treiber installieren
  - mySmartUSB im Datenmodus -> in Programmiermodus schalten (DIP)
- mySmartControl funktioniert nicht:
  - USB-Treiber (CP210x) nicht installiert -> Treiber installieren
  - kein Bootloader vorhanden -> Bootloader nachrüsten

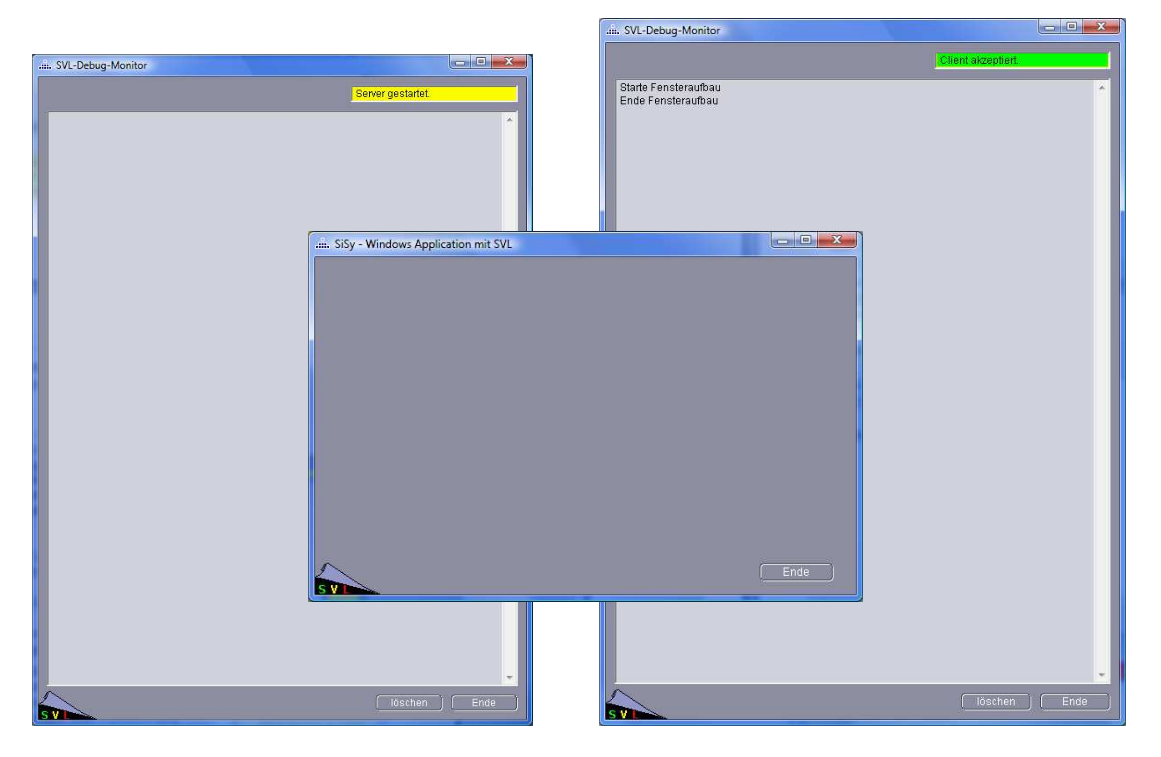
<u>Hinweis:</u>
Das myAVR ProgTool ist nur in SiSy Ausgaben mit dem Add-On AVR verfügbar.

3.8.3 SVL Werkzeuge

Der SVL-DebugMonitor

Für die Fehlersuche im laufenden Betrieb einer Windowsanwendung die auf der SVL-Klassenbibliothek basiert, steht Ihnen der SVL-Debugmonitor zur Verfügung. In der folgenden Darstellung sehen Sie eine typische Anwendung des Debugmonitors. Dabei fungiert der Debugmonitor als lokaler Server und die Anwendung als Client. Der Debugmonitor muss immer zuerst gestartet werden und danach erst die Anwendung. Sie können in der SVL Hilfe unter den Stichwort „SDebug“ detaillierte Informationen zur Nutzung der Debugschnittstelle abrufen.



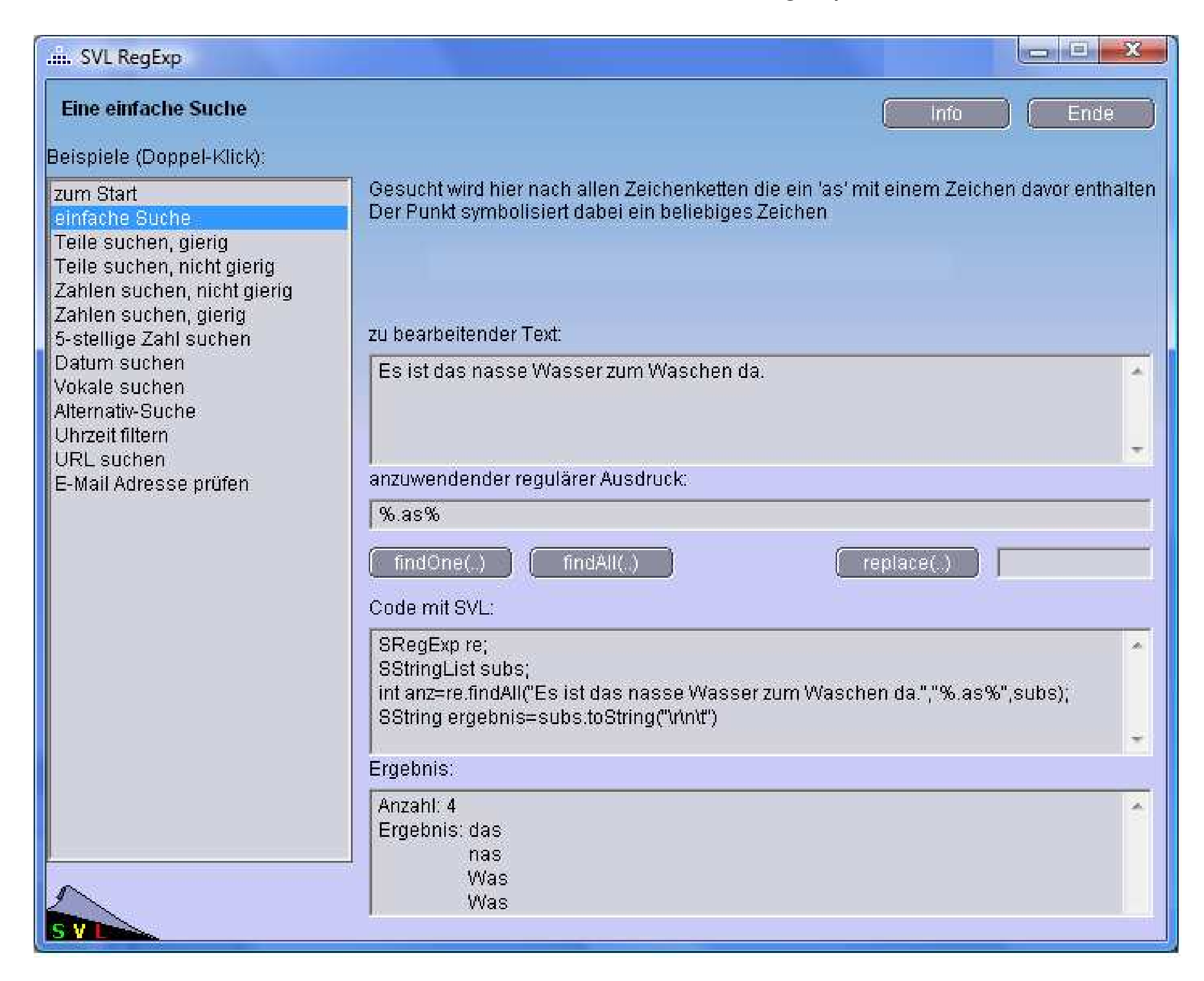


Hinweis:

SVL Anwendungen können nicht mit jeder SiSy Ausgabe erstellt werden. Sie benötigen dazu die entsprechenden Add-On's beziehungsweise die entsprechende Ausgabe von SiSy. Bei der Nutzung des Debugmonitors kann es zu Warnhinweisen Ihrer Firewall kommen. Es ist für die korrekte Funktion des Debugmonitors notwendig, die Verbindung zwischen Anwendung und Debugmonitor nicht zu blockieren.

Das SVL-Werkzeug RegExp

Reguläre Ausdrücke sind Regeln für die Suche in Zeichenketten. Man definiert also mit dieser Syntax Suchmuster. Benutzen Sie dieses Werkzeug um die Syntax eines gewünschten regulären Ausdruckes zu testen. Sie erhalten zusätzlich den C++ Code für die Anwendung des getesteten regulären Ausdrucks mit der SVL. Weitere Informationen finden Sie in der SVL-Hilfe unter den Stichwort „SRegExp“



Hinweis:

SVL Anwendungen können nicht mit jeder SiSy Ausgabe erstellt werden. Sie benötigen dazu die entsprechenden Add-On’s beziehungsweise die entsprechende Ausgabe von SiSy.

3.8.4 Weitere Werkzeuge

Entsprechend der installierten Add-On’s Ihrer SiSy Ausgabe können weitere Werkzeuge verfügbar sein.

- mySmartUSB Terminal
- mySmartUSB-Light SupportBox
- myAVR BitmapEditor
- UML Tools
- Language Tools
- Export/Import Tools
- …

4 Die Hilfefunktionen in SiSy

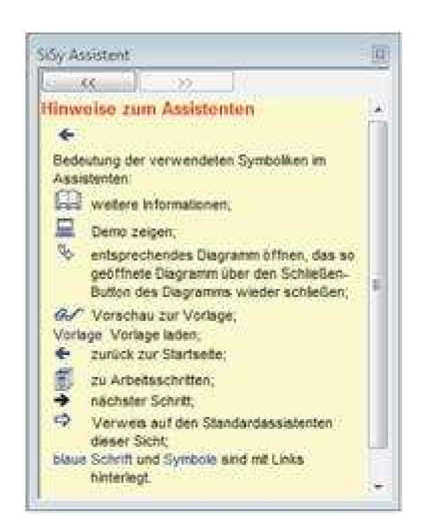
Nutzen Sie die zahlreichen Hilfen und Vorlagen, die SiSy dem Entwickler bietet!

4.1 Der Assistent

Der Assistent ist hilfreich bei der Unterstützung und Führung des Nutzers im Umgang mit SiSy. Er befindet sich standardmäßig im linken, unteren Bildschirmbereich. Der Assistent kann über die Werkzeugleiste (?-Symbol) geöffnet werden, falls dies nicht beim Programmstart erfolgte. Der Assistent begleitet Sie im gesamten Projekt. Sie erhalten immer passende Informationen zum aktuellen Diagrammtyp und haben die Möglichkeit, durch verschiedene Links weitere Hilfethemen aufzurufen oder Vorlagen in Ihr Projekt zu laden.

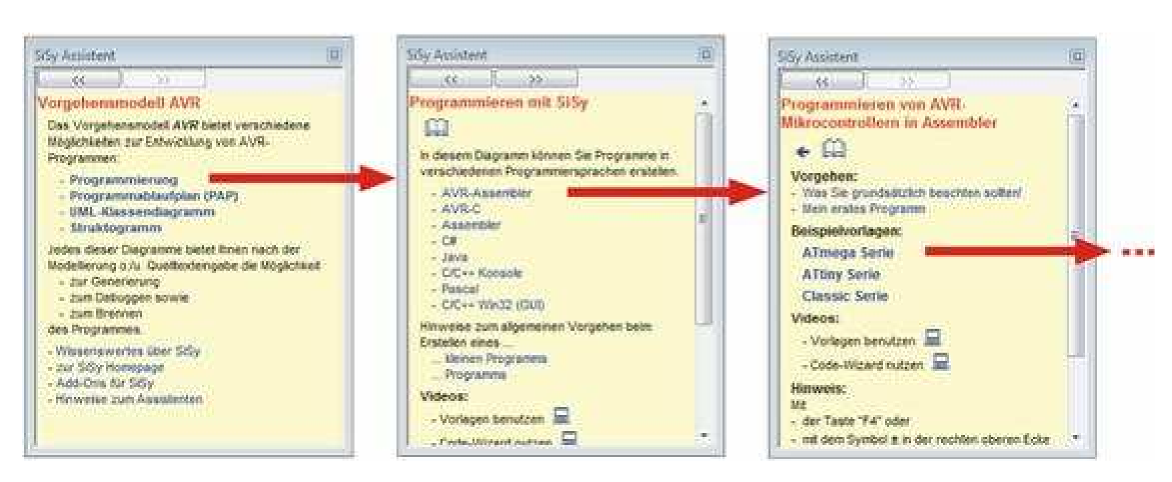
Beachte: Der Assistent ist auf die jeweilige Ausgabe von SiSy, die verfügbaren Add-Ons und das gewählte Modell bezogen.

Bedeutung der verwendeten Symboliken im Assistenten:



📖	weitere Informationen anzeigen; öffnet eine Hilfedatei (*.chm, *.hlp, *htm)
💻	Demovideo zur Handhabung zeigen; öffnet eine Animation oder Videomitschnitt der Bildschirmarbeit (AVI, ScreenCam- oder FLASH-Film)
↳	entsprechendes Diagramm öffnen, das so geöffnete Diagramm kann über die Schließen-Schaltfläche des Diagramms wieder geschlossen werden;
Beispiel 1	eine Diagrammvorlage laden;
👓	Vorschau zur Diagrammvorlage;
←	zurück zur Startseite des Assistenten;
📄	ein kleines Skript zu Arbeitsschritten anzeigen;
→	nächster Schritt (Arbeitsschritt);
⇨	Verweis auf den Standardassistenten dieser Sicht;
blaue Schrift	und
blaue Symbole	sind mit Links hinterlegt und können per Mausklick aktiviert werden.

Beispiele für Assistenten:

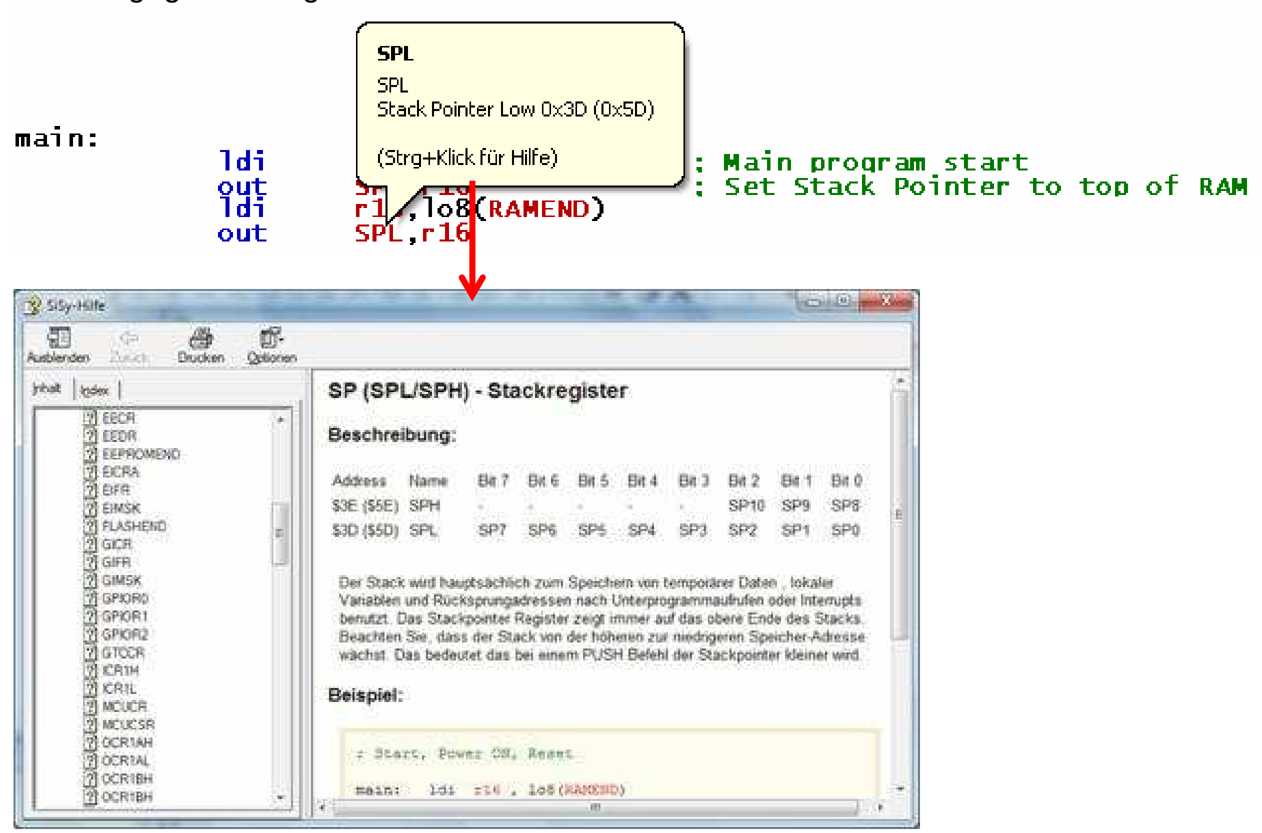


4.2 Die Online-Hilfe

Bei der Eingabe von Quellcode im dafür vorgesehenen Editorfenster werden reservierte Worte (Bezeichner, Schlüsselworte) der gewählten Programmiersprache durch verschiedenfarbiges Einfärben (Syntaxfarben) hervorgehoben. Zu den hervorgehobenen Bezeichnern existiert in der Regel auch eine kurze Online-Hilfe und eine ausführlichere Hilfe. Die Online-Hilfe ist ein Pop-Up-Fenster, welches automatisch eingeblendet wird, wenn Sie mit dem Mauszeiger auf einen Befehl zeigen. In dem Pop-Up ist eine kurze Hilfestellung zu dem Bezeichner eingeblendet. Steht eine Hilfe mit ausführlicheren Informationen zur Verfügung, wird in dem Pop-Up (STRG+Klick für Hilfe) angezeigt.

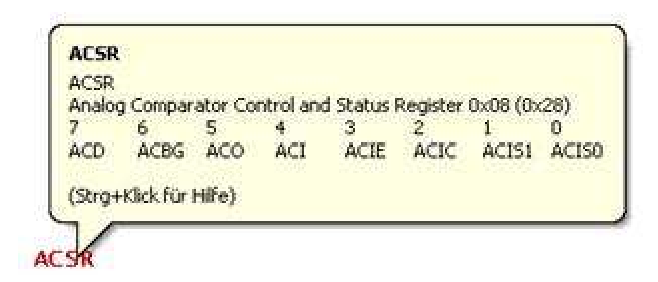
Register-Hilfe

Bei der Eingabe von bekannten Registernamen wird die Bezeichnung des Registers und dessen Adresse eingeblendet. Durch betätigen der Taste „STRG“ und gleichzeitigem Klick auf das Register öffnet eine Hilfedatei mit dem entsprechenden Hilfethema zu dem eingegeben Register.



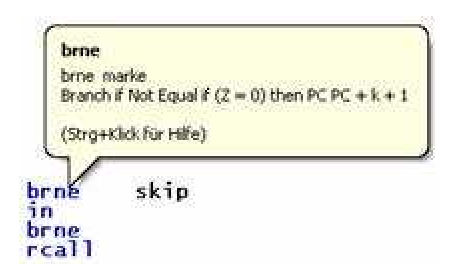
Bit-Hilfe

Wenn für ein Register detaillierte Informationen zu der Bedeutung/Funktion der einzelnen Bits vorliegen, wird zusätzlich im Pop-Up eine Kurzreferenz der Bits angezeigt. Durch drücken der Taste „STRG“ sowie Klick auf das Register öffnet SiSy eine Hilfedatei mit dem entsprechenden ausführlichen Hilfethema.



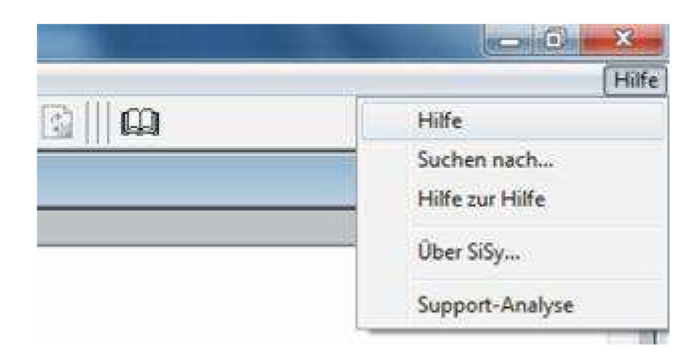
Befehlshilfe

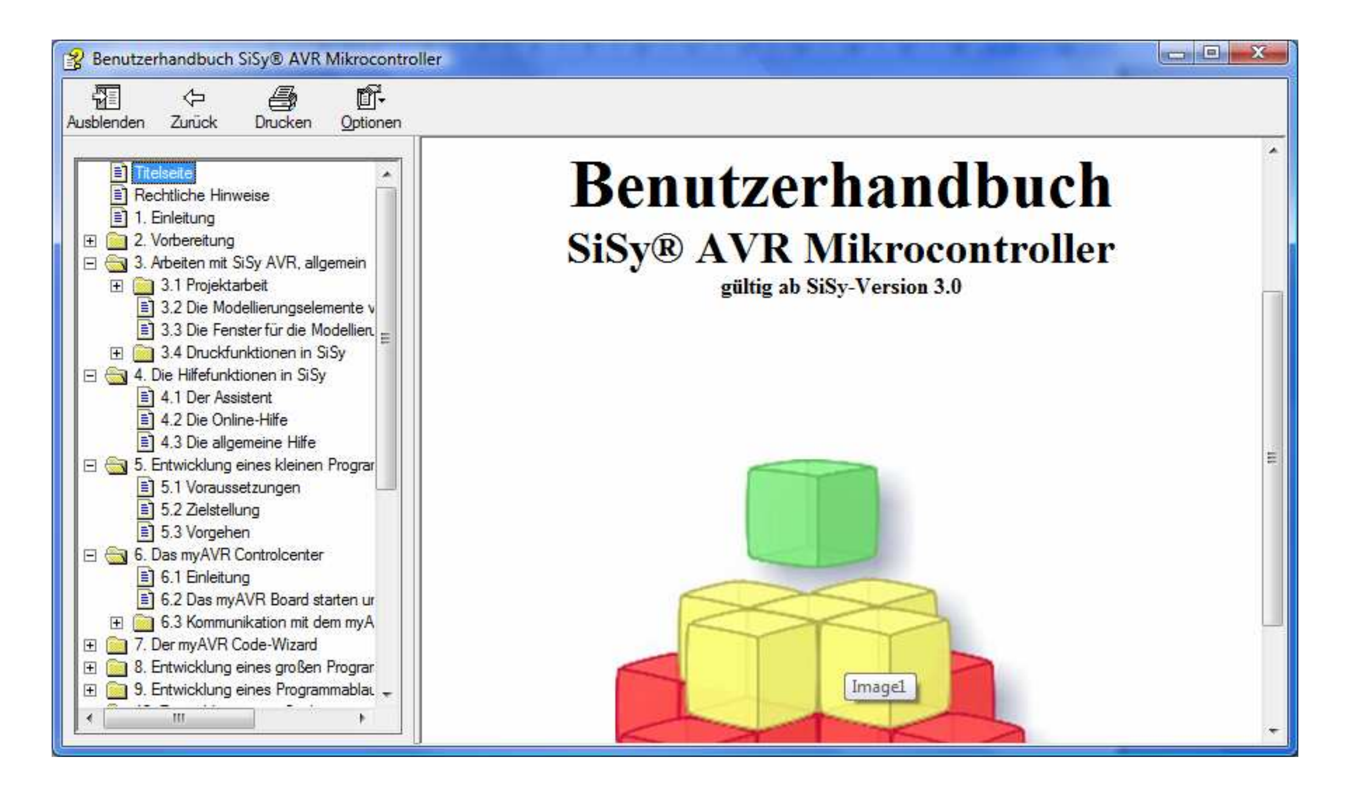
Bei der Eingabe von Befehlen wird in der Regel die Bedeutung bzw. Funktion des Befehls und ein kurzes Syntaxbeispiel eingeblendet. Durch drücken der Taste „STRG“ sowie Klick auf den Befehl öffnet SiSy eine Hilfedatei mit dem entsprechenden ausführlichen Hilfethema.



4.3 Die allgemeine Hilfe

SiSy bietet neben der direkten Hilfe bei der Eingabe von Schlüsselworten auch eine allgemeine Hilfe an.





5 Entwicklung eines kleinen Programms mit SiSy AVR

Schauen wir uns als nächstes kurz in der Entwicklungsumgebung SiSy AVR um. Für die Entwicklung von Mikrocontrollerlösungen bietet sich für den Einstieg die einfache Programmierung (kleines Programm) an.

5.1 Voraussetzungen

Für die Bearbeitung der folgenden Aufgaben benötigen Sie die aufgeführte Software und Hardware.

Software:

- SiSy AVR bzw. SiSy AVR++ ab Version 3.00

Hardware:

- Ein bestücktes myAVR Board
- Programmierkabel (USB bzw. LPT)
- eventuell Nullmodemkabel für die COM Schnittstelle
- 9 V Netzteil oder Batterie bei Bedarf (z.B.: autonomer Einsatz)
- Patchkabel

5.2 Zielstellung

Der Schnelleinstieg zur Mikrocontroller-Programmierung soll Ihnen helfen, SiSy AVR kennen zu lernen und erste Schritte in der hardwarenahen Programmierung mit SiSy zu gehen.
Zielstellung für das erste Beispielprogramm: Die drei LEDs auf dem myAVR Board sollen nacheinander aufleuchten und damit ein „Lauflicht“ erzeugen.

5.3 Vorgehen

Ein neues Projekt anlegen

Starten Sie SiSy und wählen Sie „Assistent öffnen“. Danach klicken Sie im SiSy-Assistent auf den Menüpunkt „Neues Projekt anlegen“, vergeben Sie den Projektnamen „Lauflicht“. Bestätigen Sie diesen Dialog mit „OK“. Wählen Sie als nächstes das Vorgehensmodell „Programmierung“. Es folgt ein Fenster, welches Sie mit „Weiter“ bestätigen. Als nächstes wählen Sie Ihren Programmer und Ihren Controller aus. Klicken Sie nun auf „Speichern“. Den folgenden Dialog bestätigen Sie mit „OK“. Im darauf folgenden Fenster wählen Sie die Taktrate Ihres Mikrocontrollers aus. Mit einem Klick auf „Fertig stellen“ beenden Sie dieses Fenster. Daraufhin erscheint ein Auswahlfenster. Wählen Sie „leeres Diagramm“ aus. Beenden Sie das Fenster über „Weiter“ sowie „Fertig stellen“.

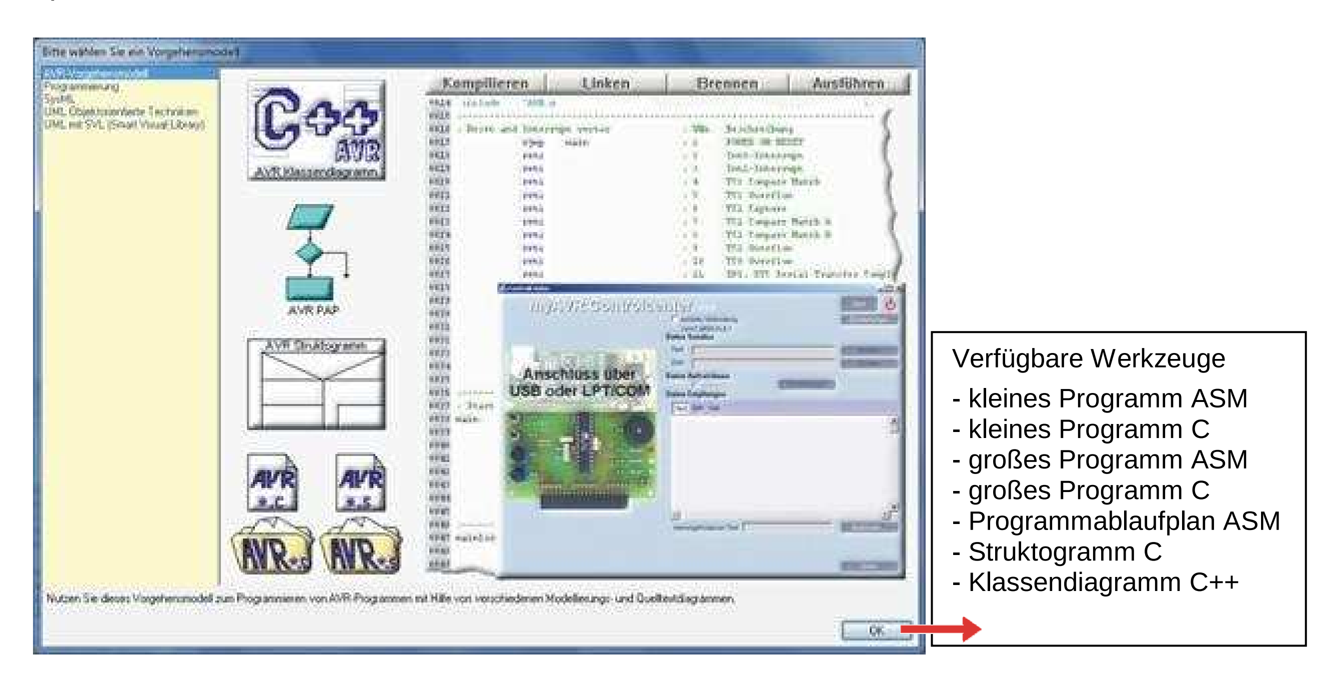


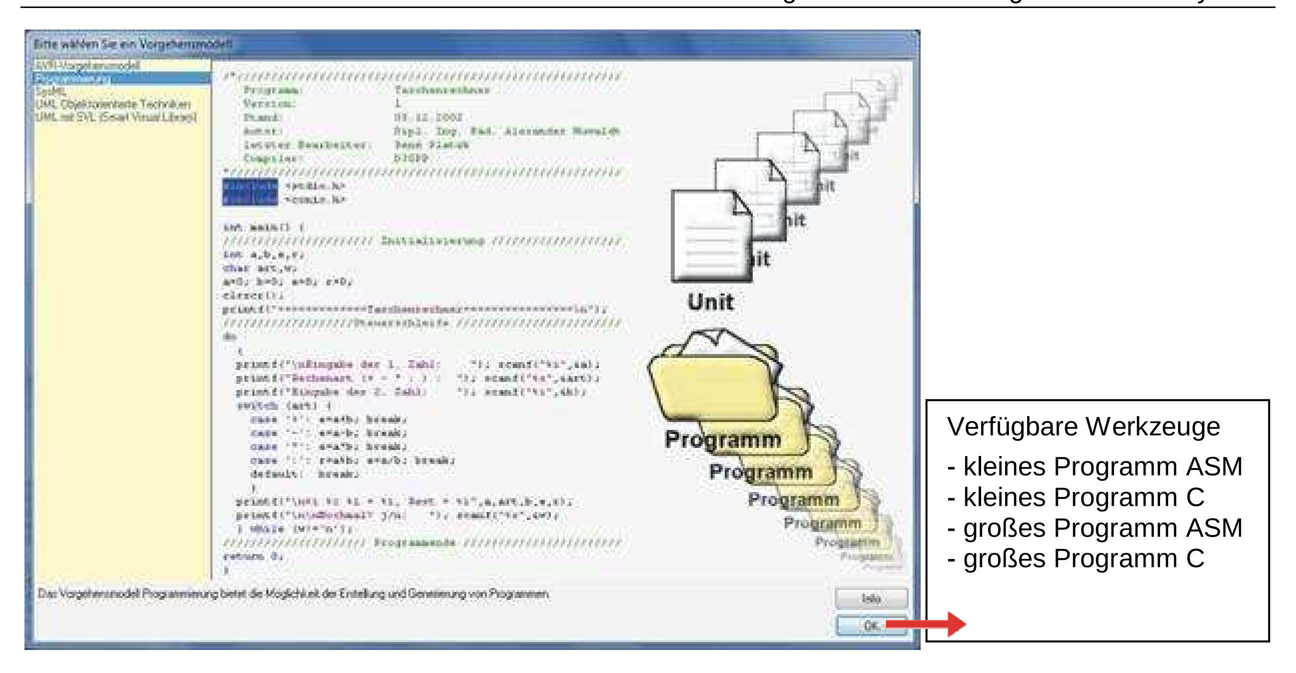
Hinweis:

In SiSy legen Sie stets ein Projekt an. In dieses Projekt integrieren Sie Ihr Programm bzw. mehrere Programme. Unabhängig vom Programmnamen benötigt jedes Projekt einen Namen.

Wenn in ihrer SiSy Ausgabe mehrere Vorgehensmodelle (Hauptebenen) zur Verfügung stehen, wird eine Auswahl der Hauptebene angeboten. Mit der Auswahl des Vorgehensmodells (VGM) entscheiden Sie über das Profil des Projektes. Wenn ein Projekt klein und die Modellierung von Programmablaufplänen nicht erforderlich ist, wählt man das einfachere Vorgehensmodell; in dem gezeigten Beispiel „Programmierung“. Bei komplexen Projekten, für die neben kleinen Programmen auch Programmablaufpläne, Struktogramme oder Klassendiagramme nötig sind, wird das komplexere Vorgehensmodell „AVR-Vorgehensmodell“ ausgewählt.

Die Auswahl des Vorgehensmodells bestimmt im weiteren Projektverlauf die von SiSy zur Verfügung gestellten Werkzeuge. Bei Auswahl des einfachen Vorgehensmodells „Programmierung“ stehen dann zum Beispiel keine grafischen Werkzeuge wie der Programmablaufplan zur Verfügung. Damit sind die Menüs und Objektbibliotheken entsprechend übersichtlicher.



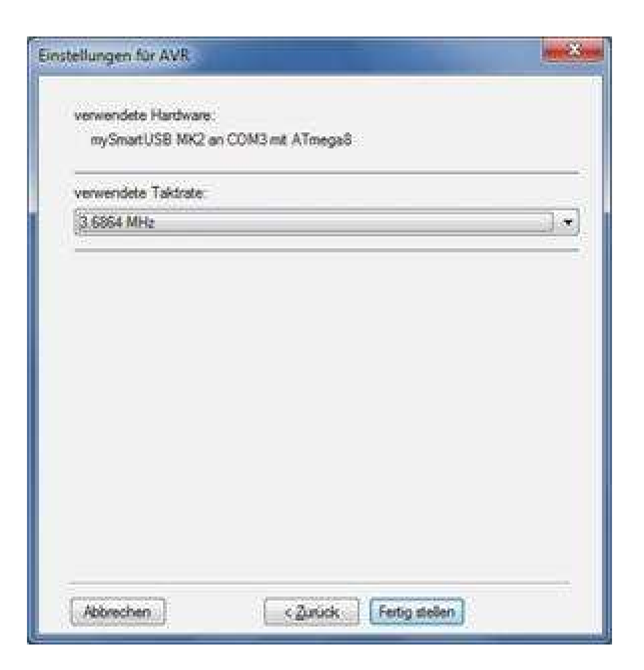


Im weiteren Verlauf werden entsprechend der Auswahl des Vorgehensmodells unterschiedliche Hilfen zum Erstellen eines Programms angeboten. Für Einsteiger empfiehlt sich zuerst die Nutzung des Vorgehensmodells „Programmierung“.
Bei jedem neuen Projekt müssen die Grundeinstellungen zur verwendeten Zielplattform vorgenommen werden (Mikrocontrollertyp, Taktrate, Programmer und I/O-Port).









Werden keine Einstellungen vorgenommen, so geht SiSy von einem mySmartUSB MK2 Programmer an COM 3 aus. Als Mikrocontroller wird ein Atmega8 mit 3,6864 MHz verwendet.

Kleines Assembler-Programm anlegen

Erstellen Sie ein Programm für den AVR-Mikrocontroller, indem Sie per Drag & Drop aus der Objektbibliothek ein „kleines Programm“ in das Diagrammfenster ziehen. Das Kontextmenü öffnet sich nun automatisch. Für spätere Bearbeitungen, markieren Sie das Objekt und wählen Sie aus dem Kontextmenü (rechte Maustaste) „Definieren“.

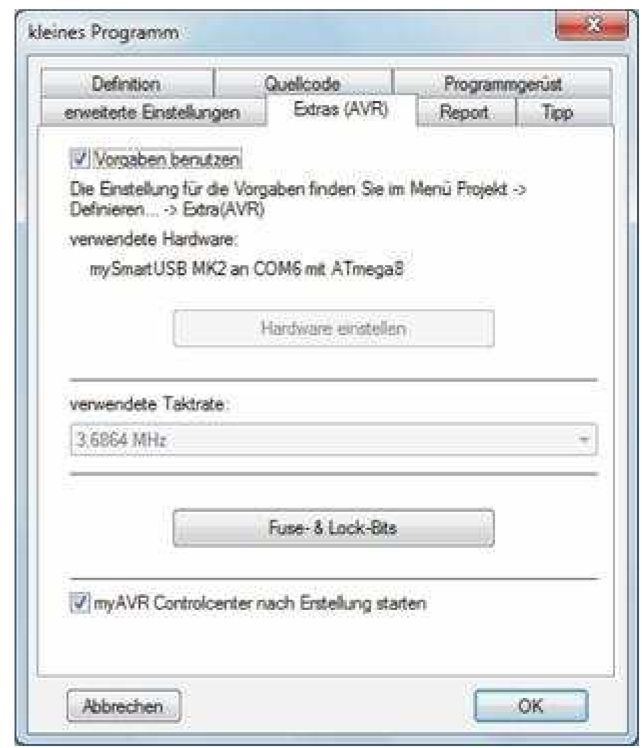
Auf der Registerkarte „Definition“ tragen Sie den Programmnamen ein (im Beispiel „Lauflicht“) und wählen die Programmiersprache aus, hier „AVR Assembler“.



Definition kleines Programm

Zielplattform und Programmer einstellen

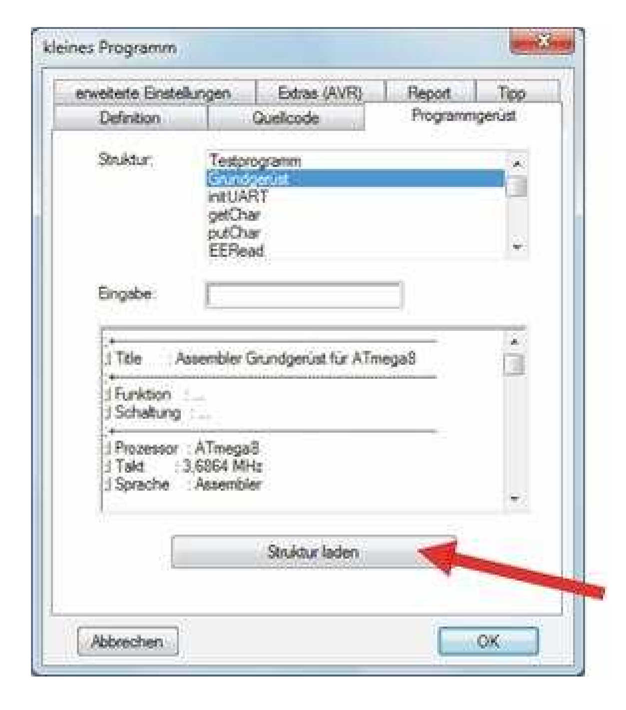
Kontrollieren Sie auf der Registerkarte „Extras (AVR)“ den ausgewählten Mikrocontroller. Die Option „Vorgaben benutzen“ überträgt automatisch die Grundeinstellungen (Mikrocontrollertyp, Taktrate, Programmer und I/O-Port) des Projektes in die lokalen Einstellungen. Sollen die lokalen Einstellungen unter „Extras (AVR)“ von den Projekteinstellungen abweichen, muss die Option „Vorgaben benutzen“ abgeschaltet werden.



Einstellungen Extras (AVR)

Programmgerüst laden, Quellcode erstellen

Über die Registerkarte „Programmgerüst" können Sie das folgende Grundgerüst für ein AVR Assemblerprogramm laden; in der Registerkarte „Quellcode" können Sie den Quellcode eigenständig eintragen.



Grundgerüst laden

Hinweis:
Den Quellcode können Sie auch im Beschreibungsfenster/Editorfenster der SiSy-Benutzeroberfläche eintragen bzw. korrigieren.

Laden Sie die Vorlage für das Programmgerüst oder erstellen Sie den folgenden Quellcode.
Vergleichen Sie dazu auch den Abschnitt zum myAVR Code-Wizard (Abschnitt 7). Das Programmgerüst darf erst geladen werden, wenn der Zielcontroller ausgewählt wurde; die Vorlagen sind controllerspezifisch.

```
;+----------------------------------------------------------------------------
;| Title                        : Assembler Grundgerüst für ATmega8
;+----------------------------------------------------------------------------
;| Prozessor                    : ATmega8
;| Takt                         : 3,6864 MHz
;| Sprache                      : Assembler
;+----------------------------------------------------------------------------
.include        "AVR.H"
;-----------------------------------------------------------------------------
begin:          rjmp   main                  ;  Power-on Reset
                reti                         ;  INT0 External Interrupt Request 0
                reti                         ;  INT1 External Interrupt Request 1
                reti                         ;  TIMER2 COMP Timer/Counter2 Compare Match
                reti                         ;  TIMER2 OVF Timer/Counter2 Overflow
                reti                         ;  TIMER1 CAPT Timer/Counter1 Capture Event
                reti                         ;  TIMER1 COMPA Timer/Counter1 Compare Match A
                reti                         ;  TIMER1 COMPB Timer/Counter1 Compare Match B
                reti                         ;  TIMER1 OVF Timer/Counter1 Overflow
                reti                         ;  TIMER0 OVF Timer/Counter0 Overflow
                reti                         ;  SPI, STC Serial Transfer Complete
                reti                         ;  USART, RXC USART, Rx Complete
                reti                         ;  USART, UDRE USART Data Register Empty
                reti                         ;  USART, TXC USART, Tx Complete
                reti                         ;  ADC ADC Conversion Complete
                reti                         ;  EE_RDY EEPROM Ready
                reti                         ;  ANA_COMP Analog Comparator
                reti                         ;  TWI 2-wire Serial Interface
                reti                         ;  SPM_RDY Store Program Memory Ready
;-----------------------------------------------------------------------------
main:           ldi         r16,hi8(RAMEND)      ; Main program start
                out         SPH,r16              ; Set Stack Pointer to top of RAM
                ldi         r16,lo8(RAMEND)
                out         SPL,r16
                ;Hier Init-Code eintragen.
;-----------------------------------------------------------------------------
mainloop:       wdr
                ;Hier den Quellcode eintragen.
                rjmp   mainloop
```

Quellcode in Assembler erstellen

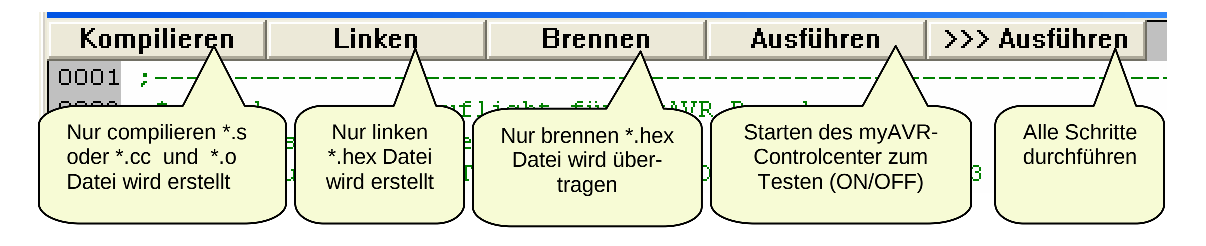
Das Lauflicht soll über die LEDs angezeigt und von dem Prozessorport D gesteuert werden. Die Realisierung erfolgt über je ein Bit im Register R18. Dieses wird mit dem Befehl Bit-Rotation nach rechts verschoben und an PORT D des Prozessors ausgegeben. Auf Grund der Prozessorgeschwindigkeit muss die Ausgabe des Lauflichtes für unser Auge verzögert werden. Geben Sie folgenden Quellcode ein bzw. ergänzen Sie die Programmvorlage!

```
;------------------------------------------------------------------------------
;* Titel              :Lauflicht für myAVR Board
;* Prozessor          :ATmega8 mit 3,6864 MHz
;* Schaltung          :PORT D.2 bis PORT D.4 an LED 1 bis 3
;* Datum              :31.01.2011
;* Autor              :Dipl. Ing. Päd. Alexander Huwaldt
;------------------------------------------------------------------------------
.include              "avr.h"
;------------------------------------------------------------------------------

; Reset and Interruptvectoren    ; VNr. Beschreibung
begin:       rjmp    main        ; 1    POWER ON RESET
             reti                ; 2    Int0-Interrupt
             reti                ; 3    Int1-Interrupt
             reti                ; 4    TC2 Compare Match
             reti                ; 5    TC2 Overflow
             reti                ; 6    TC1 Capture
             reti                ; 7    TC1 Compare Match A
             reti                ; 8    TC1 Compare Match B
             reti                ; 9    TC1 Overflow
             reti                ; 10   TC0 Overflow
             reti                ; 11   SPI, STC Serial Transfer Complete
             reti                ; 12   UART Rx Complete
             reti                ; 13   UART Data Register Empty
             reti                ; 14   UART Tx complete
             reti                ; 15   ADC Conversion Complete
             reti                ; 16   EEPROM Ready
             reti                ; 17   Analog Comparator
             reti                ; 18   TWI (I²C) Serial Interface
             reti                ; 19   Store Program Memory Redy
;------------------------------------------------------------------------------
; Start, Power ON, Reset
main:        ldi    r16 , lo8(RAMEND)
             out    SPL , r16                ; Init Stackpointer LO
             ldi    r16 , hi8(RAMEND)
             out    SPH , r16                ; Init Stackpointer HI
             ldi    r16 , 0b11111111         ; PORT D auf Ausgang
             out    DDRD , r16
             ldi    r16 , 0b00000000         ; Alle Bits auf LOW
             out    PORTD , r16
             ldi    r17 , 0b00000000
             ldi    r18 , 0b00000001         ; 1 Lauflicht-Bit
;------------------------------------------------------------------------------
mainloop:    wdr
             inc    r16                      ; Wait
             brne   skip
             inc    r17                      ; Wait
             brne   skip
             rcall  up1                      ; Lauflicht
skip:        rjmp   mainloop
;------------------------------------------------------------------------------
up1:         rol    r18                      ; Bit-Rotation
             out    PORTD , r18
             ret
;------------------------------------------------------------------------------
```

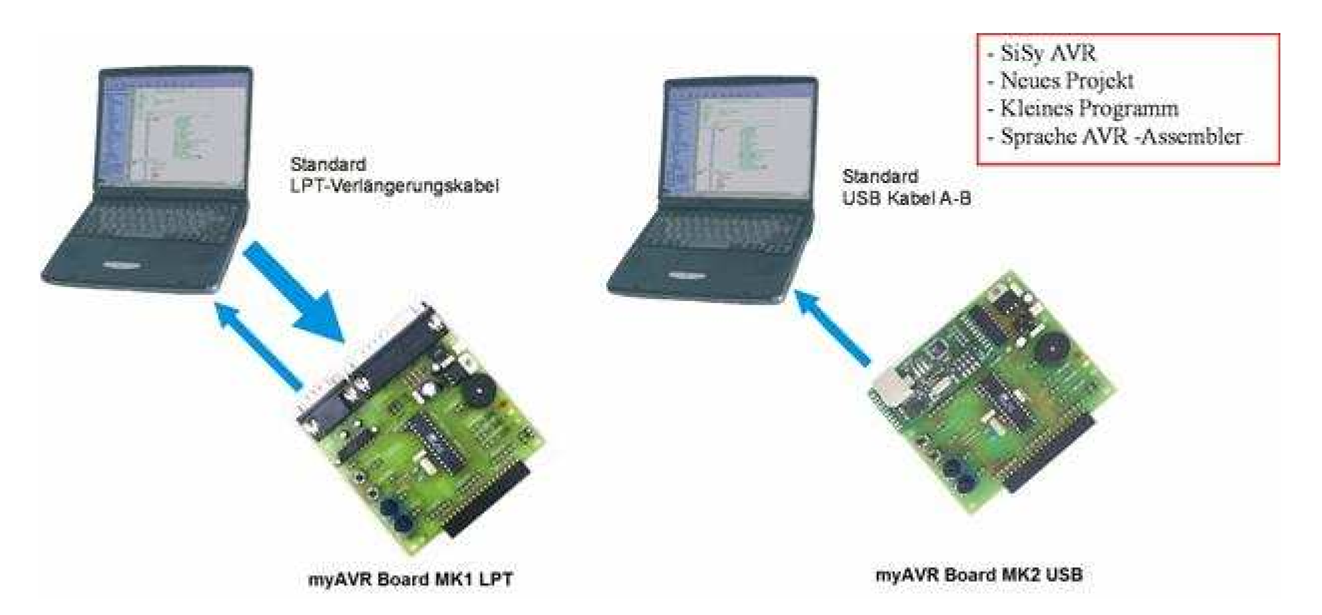

Compilieren und Linken

Der eingegebene Quellcode muss nun in Maschinencode für den AVR-Prozessor übersetzt werden. Wählen Sie dazu die Schaltflächen „Kompilieren“ und „Linken“. Bei fehlerfreier Übersetzung liegt das Programm als „Lauflicht.hex“ vor und kann auf den FLASH-Programmspeicher des Prozessors gebrannt werden.

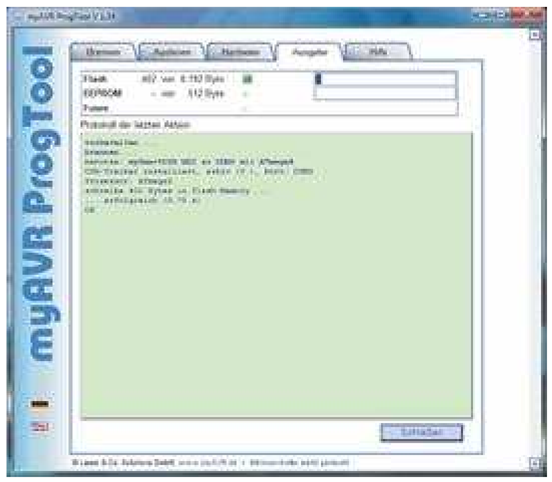


Hardware anschließen und brennen

Das myAVR Board verfügt über eine ISP (In System Programming) Schnittstelle. Der Prozessor muss also nicht für die Programmierung aus dem System entfernt werden, um ihn in einem gesonderten Programmiergerät zu brennen, sondern kann im myAVR Board direkt programmiert werden. Dazu schließen Sie entsprechend Ihrer Boardvariante das Programmierkabel an den LPT- oder USB-Port Ihres Rechners an.

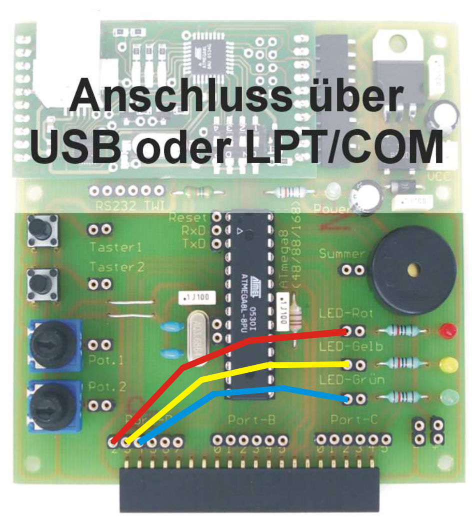


Zum Brennen wählen Sie die Schaltfläche „Brennen“. Bei erfolgreichem Brennvorgang erhalten Sie im Ausgabefenster vom myAVR ProgTool folgende Meldung:



Ausgabefenster myAVR ProgTool

Mikrocontrollerlösung testen



Verkablung

Um das Programm zu testen ist es nötig, den Port D mit den LEDs zu verbinden.

- Wenn vorhanden, ziehen Sie die Batterie bzw. das Netzteil und das Programmierkabel ab.
- Verbinden Sie die LEDs mit dem Prozessorport D entsprechend dem folgenden Schema.
  Nutzen Sie die Patchkabel!
- Prüfen Sie die Verbindungen und schließen Sie die Batterie/das Netzteil oder das Programmierkabel wieder an und nehmen Sie die Mikrocontrollerlösung in Betrieb.
- Es ist jetzt an den LEDs ein Lauflicht zu sehen.
- Gratulation!
  Das ist Ihre erste Mikrocontrollerlösung mit dem myAVR Board.

Beim Kompilieren, Linken und Brennen des Schnellstart-Beispiels öffnet sich ein Ausgabefenster und zeigt Protokollausgaben der Aktionen an. Beim Brennen öffnet sich zusätzlich das myAVR ProgTool. Wenn die Hardware ordnungsgemäß angeschlossen, von der Software erkannt und das Programm erfolgreich auf den Programmspeicher des Mikrocontrollers übertragen wurde, schließen Sie das myAVR ProgTool. Die letzte Ausschrift hat folgenden Inhalt:

```
brenne Daten neu
Ende.
```

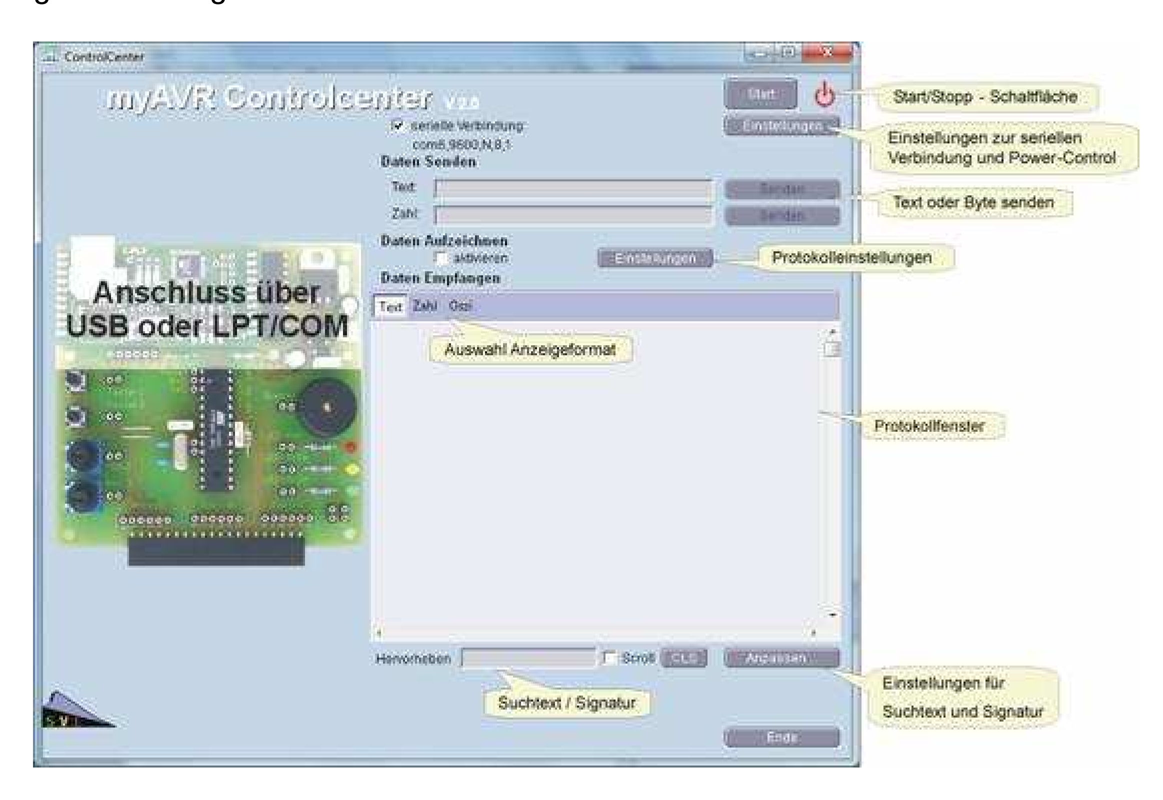
Ausgabefenster mit "Brenn"-Protokoll

Inbetriebnahme, Test und Datenkommunikation mit der Mikrocontrollerlösung erfolgen über das myAVR Controlcenter. Dabei wird über die Schaltfläche „Start“ das Board mit der nötigen Betriebsspannung versorgt und der Controller gestartet. Der Datenaustausch mit dem myAVR Board ist möglich, wenn das Null-Modemkabel an Rechner und Board angeschlossen ist, sowie die Mikrocontrollerlösung dafür vorgesehen ist. Es können Texte und Bytes (vorzeichenlose ganzzahlige Werte bis 255) an das Board gesendet und Text empfangen werden. Die empfangenen Daten werden im Protokollfenster angezeigt. Vergleichen Sie dazu den folgenden Abschnitt zum myAVR-Controlcenter.

6 Das myAVR Controlcenter

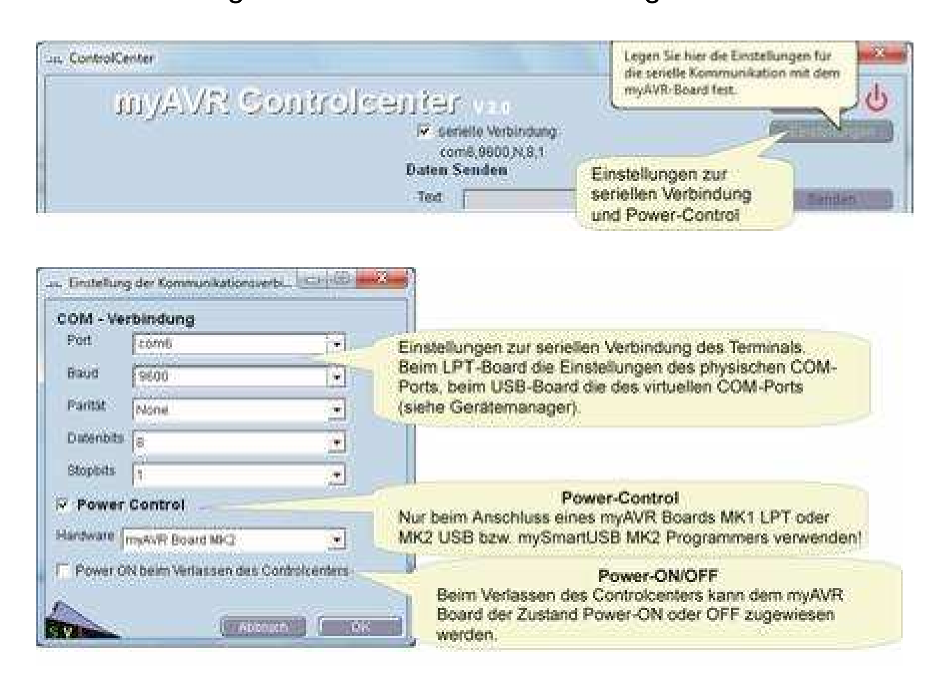
6.1 Einleitung

Das myAVR Controlcenter ist ein universelles Terminalprogramm zur Kommunikation mit dem myAVR Board und anderen Mikrocontrollerapplikationen, die über eine serielle Schnittstelle (UART) oder USB Anbindung mit virtuellem COM-Port zum PC verfügen. Es kann für Test- und Debug-Meldungen sowie Visualisierung und Protokollierung von Messdaten genutzt werden. Dazu bietet das myAVR Controlcenter umfangreiche Konfigurationsmöglichkeiten.



6.2 Das myAVR Board starten und stoppen (Power Control)

Wenn Sie ein myAVR Board MK1 LPT oder MK2 USB besitzen, können Sie die Spannungsversorgung des Boards aus dem Controlcenter heraus steuern. Dazu wählen Sie die Einstellungen für die serielle Verbindung und Power-Control.



Mit dem Aktivieren der Power-Control-Funktion wird beim Betätigen der Schaltfläche „Start/Stopp“ die rechnerseitige Spannungsversorgung des myAVR Boards und die RESET-Leitung angesteuert.

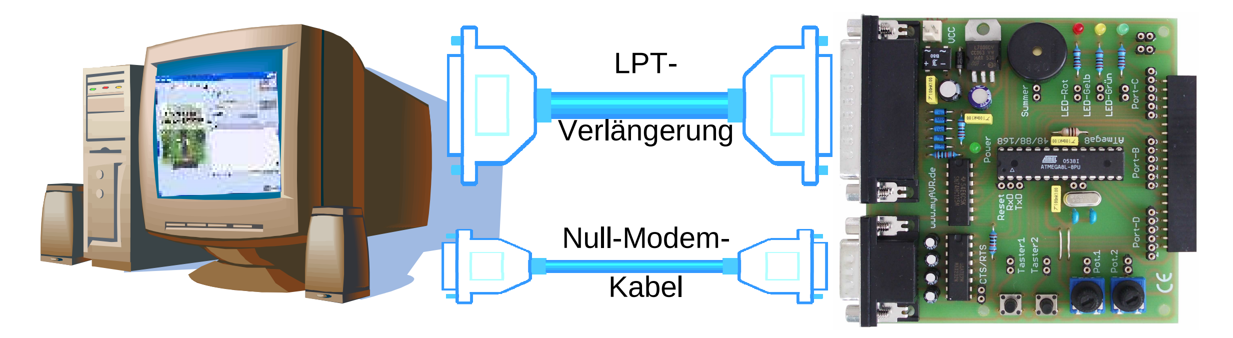


Bei einer externen Spannungsversorgung, zum Beispiel durch einen 9V Block oder das myAVR Netzteil, hat die Power-Control-Funktion keine Wirkung auf die Boardspannung. In diesem Fall bezieht sich die Wirkung ausschließlich auf die RESET-Leitung, um den Controller zu starten bzw. zu stoppen. Beachten Sie, dass beim myAVR Board MK1 LPT die Spannungsversorgung über den LPT-Port für eine serielle Kommunikation und den Betrieb des LC-Displays nicht ausreicht, dafür wird die Verwendung eines myAVR Netzteils empfohlen.

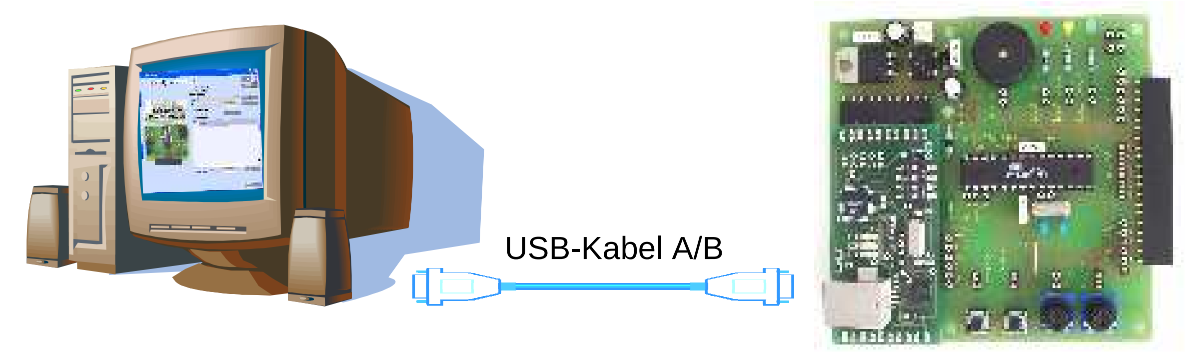
6.3 Kommunikation mit dem myAVR Board

6.3.1 Grundlagen (LPT/USB-Variante)

Die zwei myAVR Boardversionen unterscheiden sich grundsätzlich in der technischen Realisierung der seriellen Kommunikation. Das myAVR Board MK1 LPT verfügt über eine gesonderte RS232 Schnittstelle (COM-Port) und wird über ein Nullmodemkabel an einen physisch vorhandenen COM-Anschluss des PC angeschlossen.

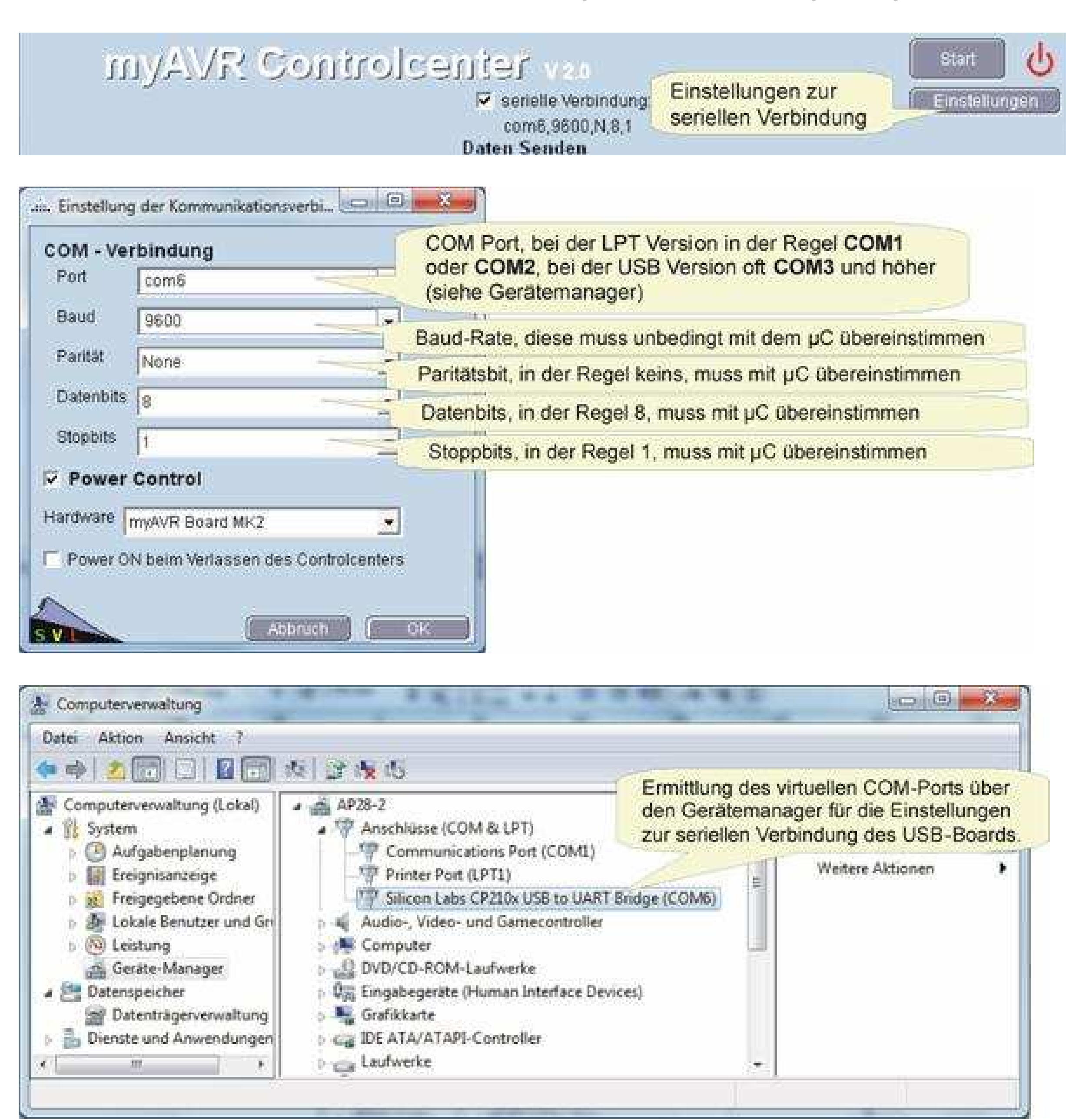


Das myAVR Board MK2 USB verfügt über den USB Programmer mySmartUSB MK2. Dieser stellt gleichzeitig einen virtuellen COM-Port für die Kommunikation zur Verfügung.



6.3.2 Einstellungen für die seriellen Verbindung

Für eine erfolgreiche Kommunikation mit dem myAVR Board ist es wichtig, dass Sender und Empfänger von seriellen Daten die gleichen Parameter für die Datenübertragung konfiguriert haben. Auf der PC Seite werden die Kommunikationsparameter im Controlcenter über die Schaltfläche für die Einstellungen der Verbindung konfiguriert.



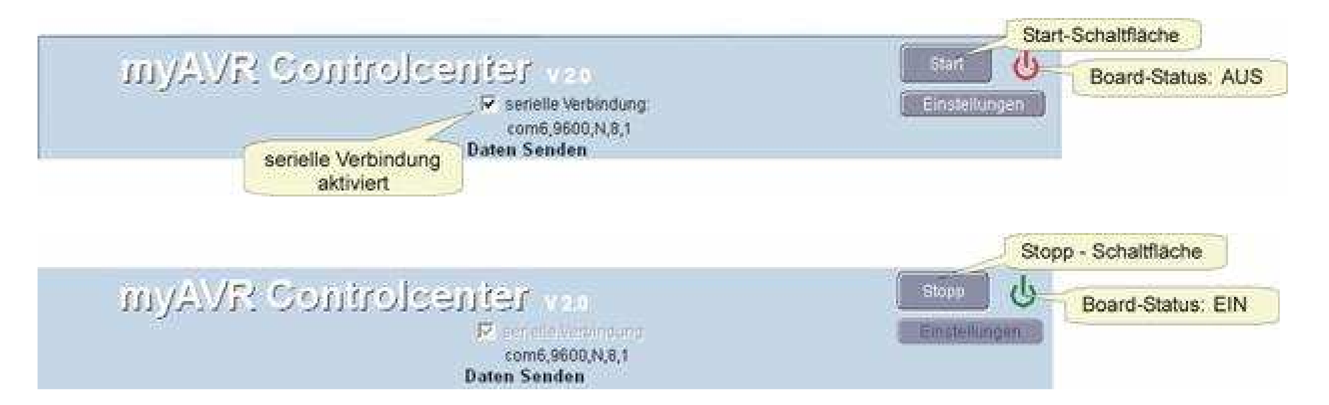
Im Mikrocontrollerprogramm sind die gleichen Parameter bei der Initialisierung der UART zu wählen. Beachten Sie die eingestellte Taktquelle (hier Quarz mit 3,6 MHz).

```
#define     F_CPU    3686400
#define     BAUD     9600
#include    <avr\io.h>
//-------------------------------------
// UART initialisieren
void initUART()
{
    sbi(UCSRB,3);    // TX aktiv
    sbi(UCSRB,4);    // RX aktivieren
    UBRRL=(uint8_t)(F_CPU/(BAUD*16L))-1;        // Baudrate festlegen
    UBRRH=(uint8_t)((F_CPU/(BAUD*16L))-1)>>8;   // Baudrate festlegen
}
```

Beispiel für die Konfiguration der UART des Mikrocontrollers (µC) mit 9600 Baud.

6.3.3 Daten empfangen vom myAVR Board

Das myAVR Controlcenter empfängt Daten über den gewählten COM-Port und stellt diese im Protokollfenster dar. Damit können Statusmeldungen, Fehlermeldungen oder auch Messwerte erfasst werden. Die Voraussetzung ist, dass die serielle Verbindung hergestellt (Nullmodemkabel oder USB Kabel), korrekt konfiguriert und aktiviert wurde. Die Kommunikation beginnt mit dem Betätigen der Schaltfläche „Start“ und endet mit dem Betätigen der Schaltfläche „Stopp“. Der Zustand (in Betrieb/Halt) wird über das Symbol (rot/grün, EIN/AUS) rechts neben der Schaltfläche angezeigt.



6.3.4 Darstellung der empfangen Daten

Die empfangenen Daten werden fortlaufend im Protokollfenster dargestellt. Der Darstellungsmodus kann während der Kommunikation umgeschaltet werden. Das Controlcenter bietet folgende Darstellungsmodi:

- Text
- Zahl
  - Dezimal
  - Hexadezimal
- Grafik (Oszi)

Die Daten des folgenden Programmbeispiels sollen als Testdaten dienen:

```
//----------------------------------------------------
main()
{
    uint8_t a='A';                      // Zeichen
    initUART();                   // Initialisierungen
    while (true)                  // Mainloop-Begin
    {
        putChar(a);                     // Zeichen senden
        a++;                            // nächstes Zeichen
        myWait_1000ms();                // Pause
    }                             // Mainloop-Ende
}
//----------------------------------------------------
```

Der Textmodus

Der Textmodus dient zur Visualisierung alphanumerischer Werte im ASCII Format (Zeichenketten). Zahlen, die mit Zeichenketten gesendet werden, müssen vom Mikrocontrollerprogramm zuvor ins ASCII-Format gewandelt werden (siehe `itoa` und `sprintf`).

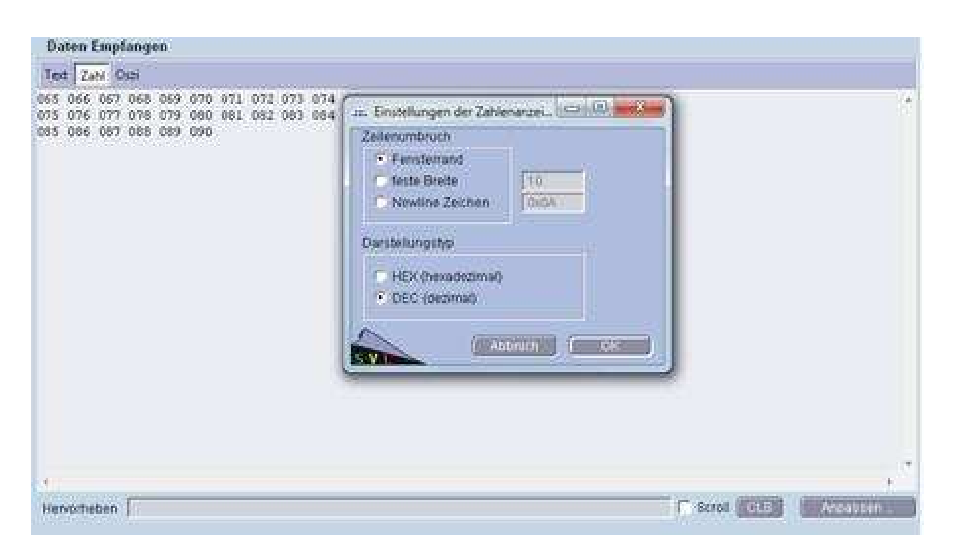


Über die Schaltfläche „Anpassen " lässt sich das Protokollfenster weiter konfigurieren. Die Zeilenbreite und das Zeichen für einen Zeilenumbruch lassen sich auswählen. Es können wichtige/gesuchte Textstellen im Protokollfenster hervorgehoben werden.

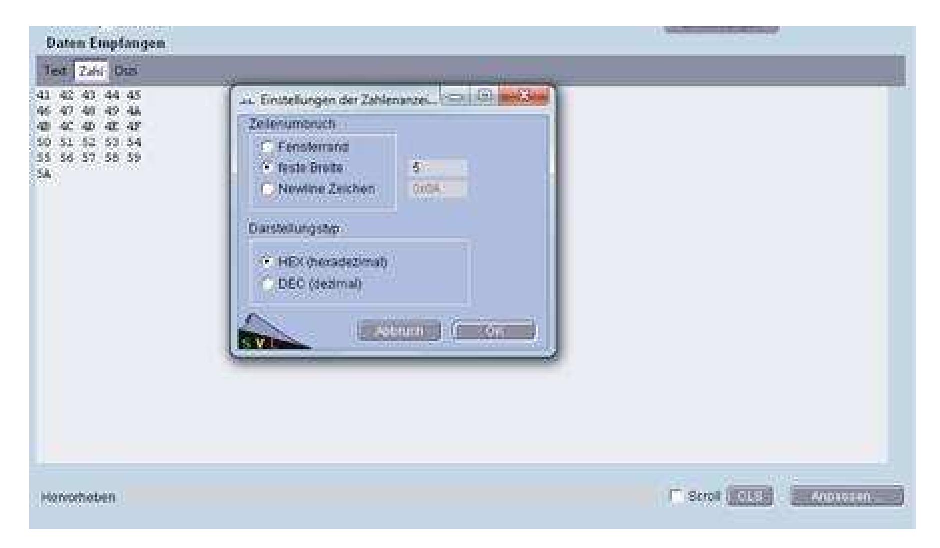


Der Zahlenmodus

Der Zahlenmodus visualisiert die empfangenen Datenbytes (8 Bit, Werte von 0 bis 255) als Dezimal- oder Hexadezimalzahlen. Dezimalzahlen werden dreistellig mit führender Null dargestellt.



Die Anzeige kann über die Schaltfläche „Anpassen“ weiter konfiguriert werden. Dabei werden die Zeilengröße oder das Zeilenumbruchzeichen sowie das Format der Zahlendarstellung (Hexadezimal/Dezimal) ausgewählt.

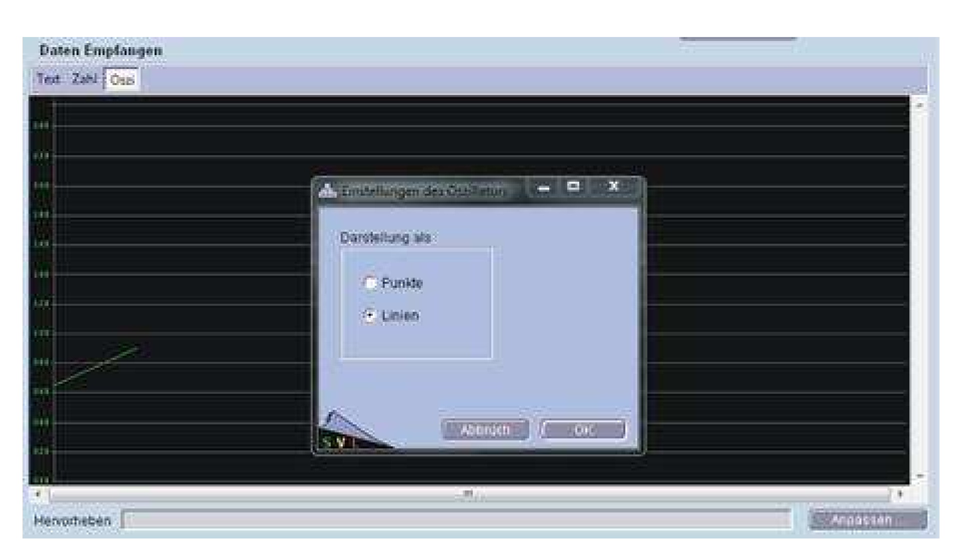


Es können wichtige/gesuchte Zahlenwerte im Protokollfenster hervorgehoben werden. Hexadezimale Zahlen sind durch den Präfix 0x zu kennzeichnen. Mehrere Werte können mit Leerzeichen getrennt angegeben werden.



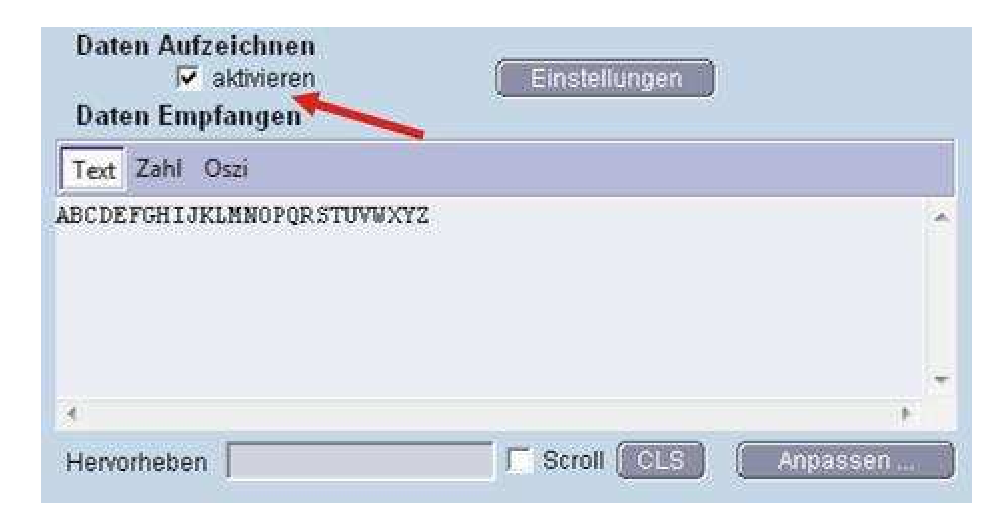
Der Grafikmodus

Messwerte können auch grafisch visualisiert werden. Dabei werden die Werte fortlaufend und byteweise (Wertebereich 0 bis 255) als Punkte in einem Koordinatensystem visualisiert. Die X-Achse repräsentiert den zeitlichen Verlauf, die Y-Achse den Wertebereich der Daten. Die Anzeige kann als einzelne Punkte oder Linien erfolgen.

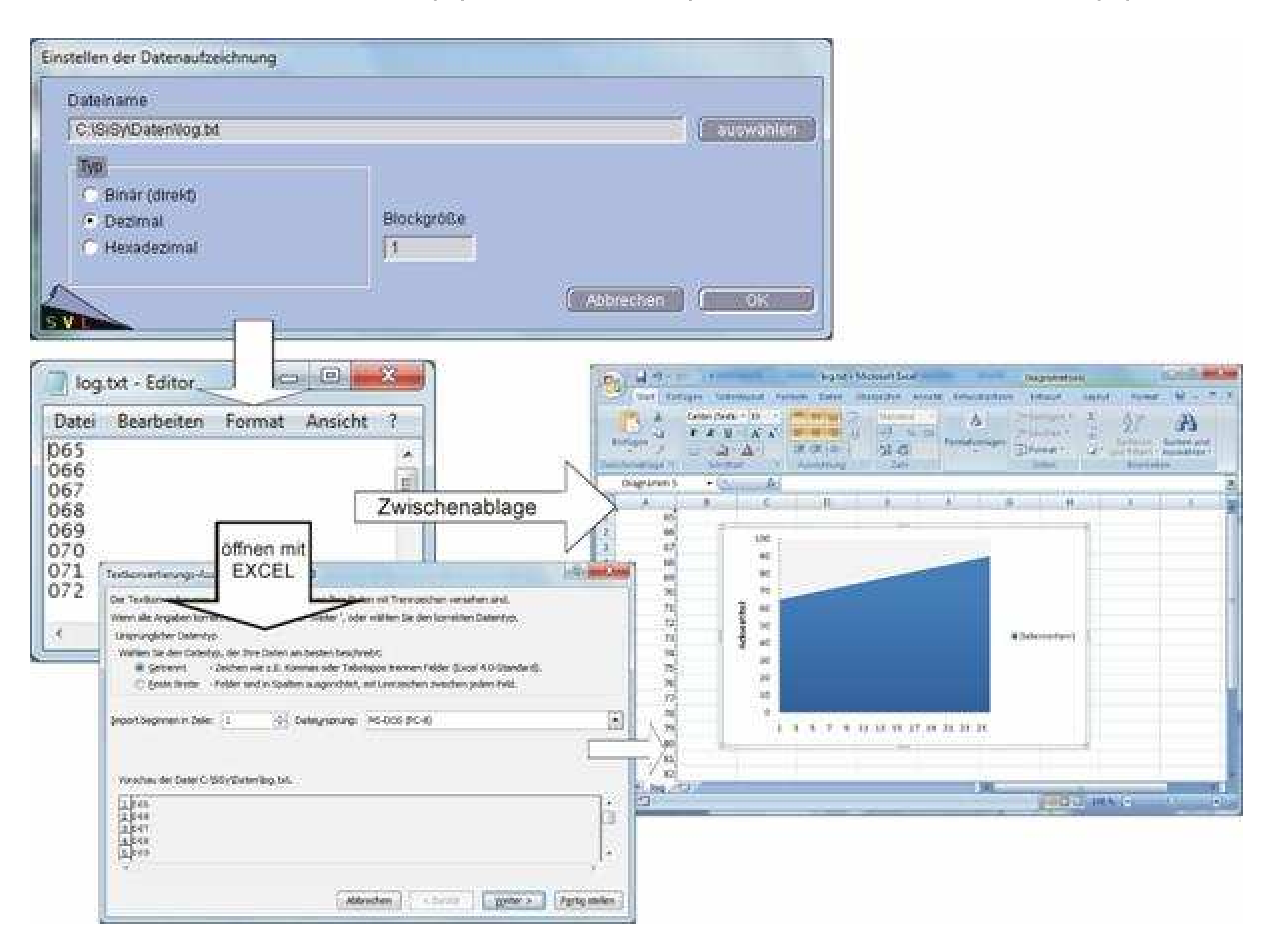


6.3.5 Empfangene Daten speichern

Bei der Erfassung von Messdaten die vom Mikrocontroller an den PC gesendet werden ist es oft wichtig, diese in eine Datei zu speichern. Damit wird eine Weiterverarbeitung der Daten in entsprechenden Programmen (z.B.: EXCEL) möglich. Das myAVR Controlcenter ermöglicht es, Protokolldaten aufzuzeichnen (Log-Datei, Rekorderfunktion). Um diese Funktion zu aktivieren, wählen Sie die Option „Daten aufzeichnen“



Über die Schaltfläche „Einstellungen“ muss der Dateiname und Pfad der Log-Datei angegeben werden. Zusätzlich ist es möglich, das Format und die Blockgröße (Zeilenumbruch) festzulegen. Die Weiterverarbeitung der Daten erfolgt entsprechend der Möglichkeiten der Zielanwendung (z.B.: Öffnen, Importieren oder Zwischenablage).



6.3.6 Daten an das myAVR Board senden

Über das Controlcenter können Daten im Text- oder Zahlenformat an den Mikrocontroller gesendet werden. Das folgende Programm dient als Veranschaulichung dieser Funktion.

```
//---------------------------------------------------------
main()
{
    uint8_t zeichen;
    initUART();                 // Initialisierungen
    while (true)                // Mainloop-Begin
    {
        zeichen=getChar();      // Zeichen abholen
        putChar(zeichen);       // Zeichen zurücksenden
    }                           // Mainloop-Ende
}
//---------------------------------------------------------
```

Text senden

Es können einzelne Zeichen, aber auch Zeichenketten gesendet werden. Mit der zugehörigen Schaltfläche „Senden“ wird immer der gesamte Inhalt der Eingabezeile „Text“ gesendet.



Zahlen senden

Zahlenwerte von 0 bis 255 (1 Byte) können einzeln oder als Zahlenfolge gesendet werden. Zwischen den Werten ist ein Leerzeichen als Trennzeichen einzufügen. Zahlen, die größer sind als 255, werden auf den niederwertigen Teil gekürzt. Es ist möglich, Zahlen im Hexadezimalformat zu schreiben (Präfix 0x). Mit der zugehörigen Schaltfläche „Senden“ wird immer der gesamte Inhalt der Eingabezeile „Zahl“ gesendet.



7 Der myAVR Simulator MK1/2

7.1 Einleitung

Mikrocontrolleranwendungen werden zur Laufzeit auf dem entsprechenden Zielsystem verarbeitet. Nicht jede Zielhardware verfügt über eine eigne Debughardware. Für das myAVR Board MK1 und MK2 mit dem ATmega8 bietet der myAVR Simulator MK1/2 die Möglichkeit, Assemblerprogramme auf einer virtuellen Hardware zu simulieren. Der Simulator ist vor allem zu Lernzwecken und für Einsteiger konzipiert und verfügt nicht über alle Möglichkeiten der kompletten AVR-Controllerfamilie.

Anzeige der benutzten Arbeitsregister

Schaltflächen für: ASM-Datei laden und Zusatzfenster öffnen

Textfeld zur Eingabe von Registerwerten

Schaltflächen zur Steuerung

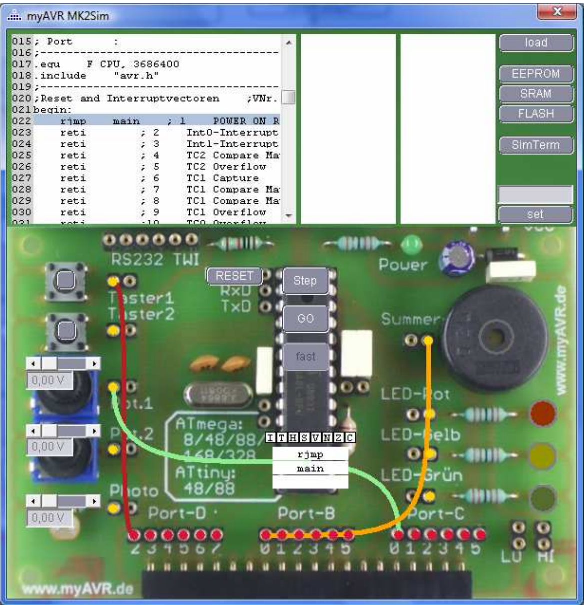
Eingabe: Taster Poti Lichtsensor

Ausgabe: Speaker LEDs

Flags und aktueller Befehl

Port-Pins des ATmega8

7.2 Den Simulator starten und konfigurieren

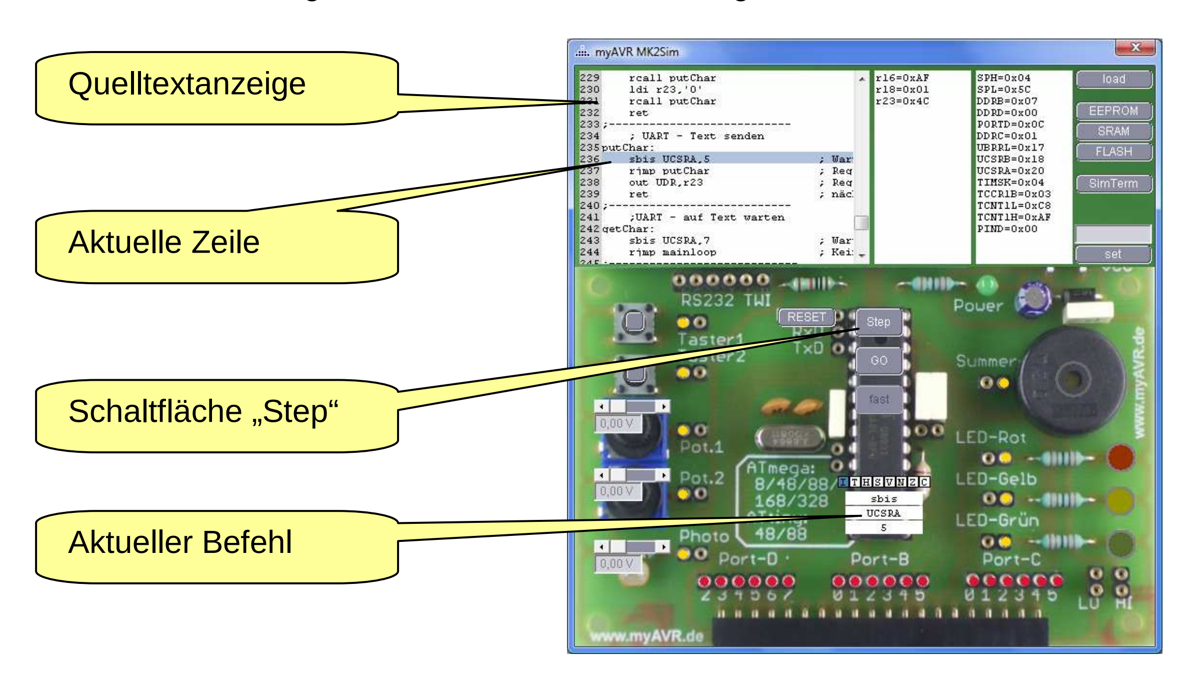


Der Simulator kann über das Menü *Werkzeuge* oder aus dem Diagrammfenster für die Programmierung von AVR Assemblerprogrammen gestartet werden. Wird der Simulator aus dem Menü Werkzeuge gestartet, muss die gewünschte Assemblerdatei vor der Simulation geladen werden. Als nächstes sind die Hardwareverbindungen zu Ein- und Ausgabegeräten herzustellen. Die Anschlusspunkte sind dazu jeweils nacheinander mit der Maus anzuklicken. Mit der rechten Maustaste kann Farbe und Form der virtuellen Patchkabel geändert werden.

7.3 Die Programmsimulation durchführen

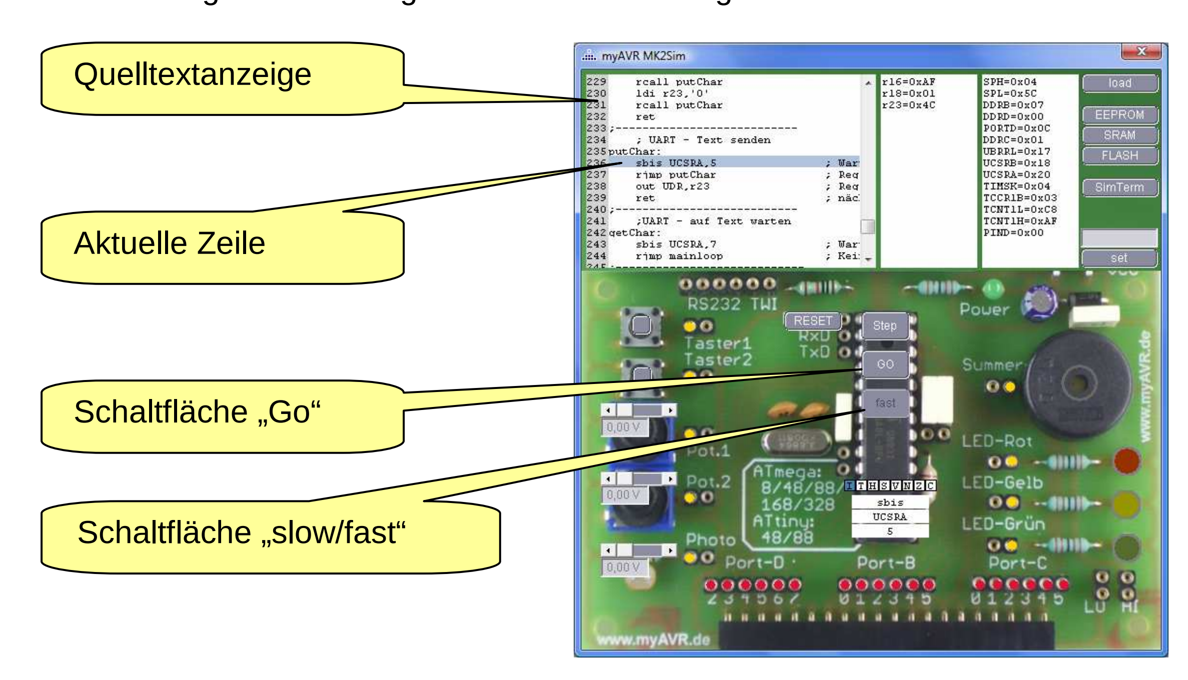
7.3.1 Einzelschritte

Die einfachste Form der Simulation ist der Einzelschrittbetrieb. Dabei wird Anweisung für Anweisung ausgeführt. Die Schaltfläche „Step" simuliert dabei das Taktsignal des Controllers. Die Abarbeitung eines Befehls erfordert eine dem realen Befehl entsprechende Anzahl von Steps (Takten, vgl. Datenblatt des ATmega8). Die meisten Befehle des AVR RISC-Core benötigen einen Takt zur Abarbeitung.



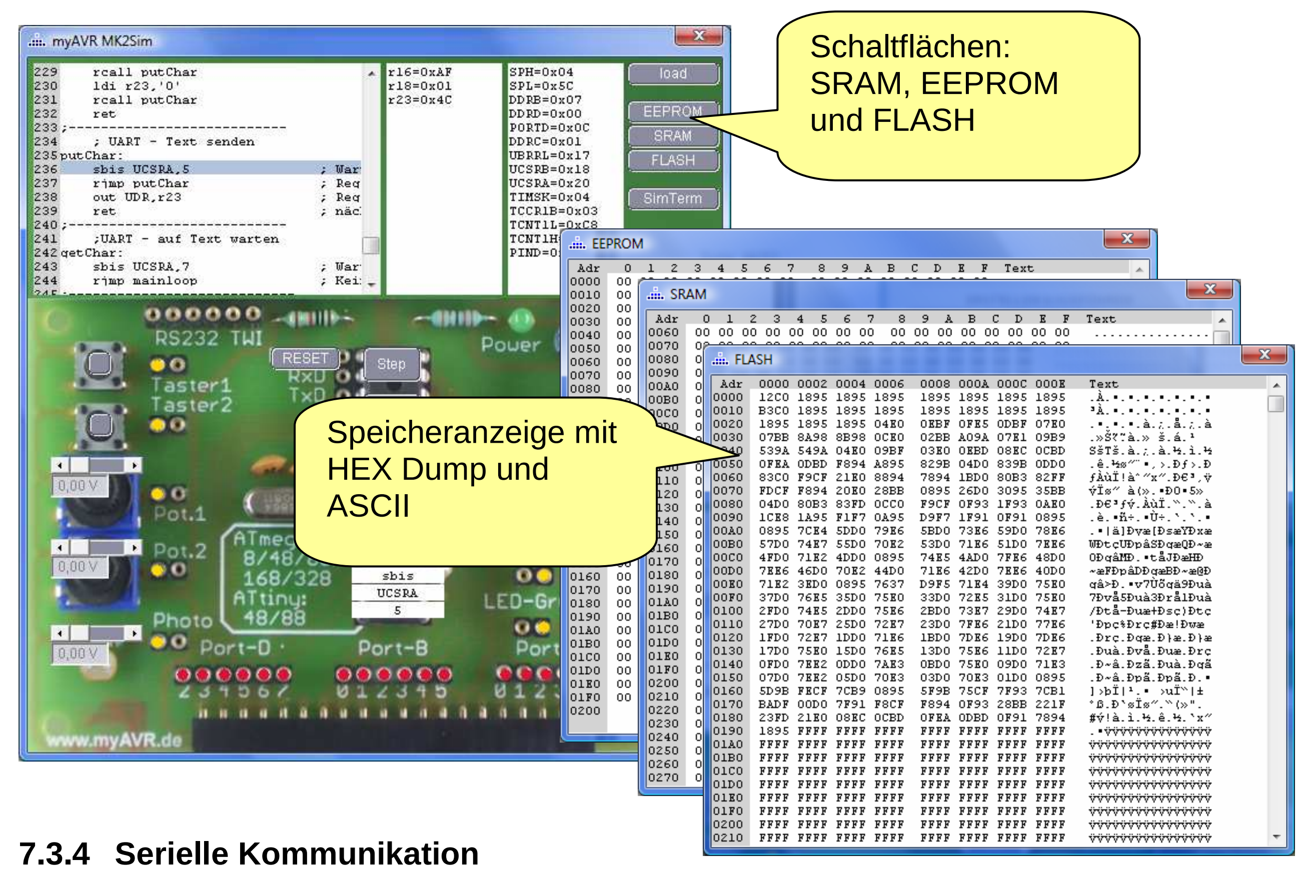
7.3.2 Langsame und schnelle Simulation

Die Schaltfläche „Go" simuliert einen kontinuierlichen Takt. Es ist nicht mehr nötig, die Schaltfläche „Step" zu betätigen. Mit der Schaltfläche „slow/fast" kann die Simulationsgeschwindigkeit umgeschaltet werden. Die Quellcodeanzeige und die aktuelle Programmzeile werden während der schnellen und langsamen Simulation nachgeführt, jedoch kann das Auge nur der langsamen Simulation folgen.



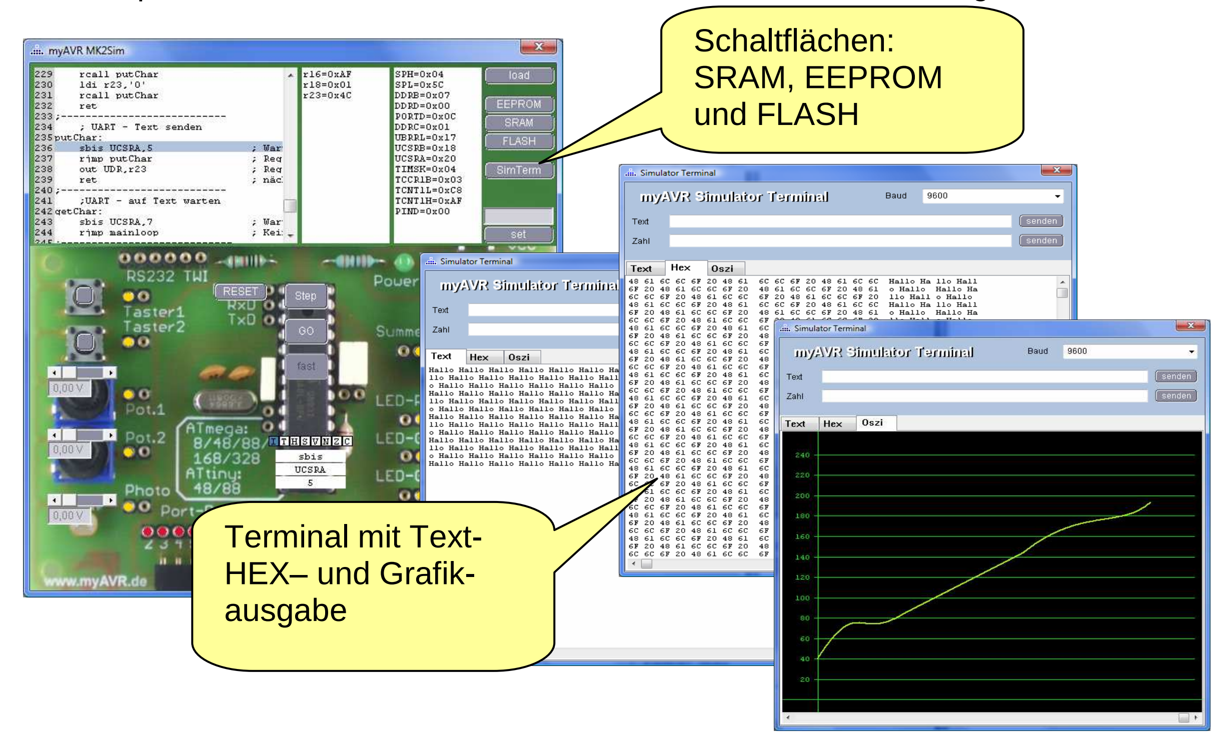
7.3.3 Speicherinhalte anzeigen

Es ist möglich im Schrittbetrieb oder während der automatischen Simulation die Speicherinhalte von FLASH, EEPROM und SRAM zu betrachten und Änderungen zu verfolgen. Die Speicheranzeigen öffnen Sie mit den entsprechenden Schaltflächen.



7.3.4 Serielle Kommunikation

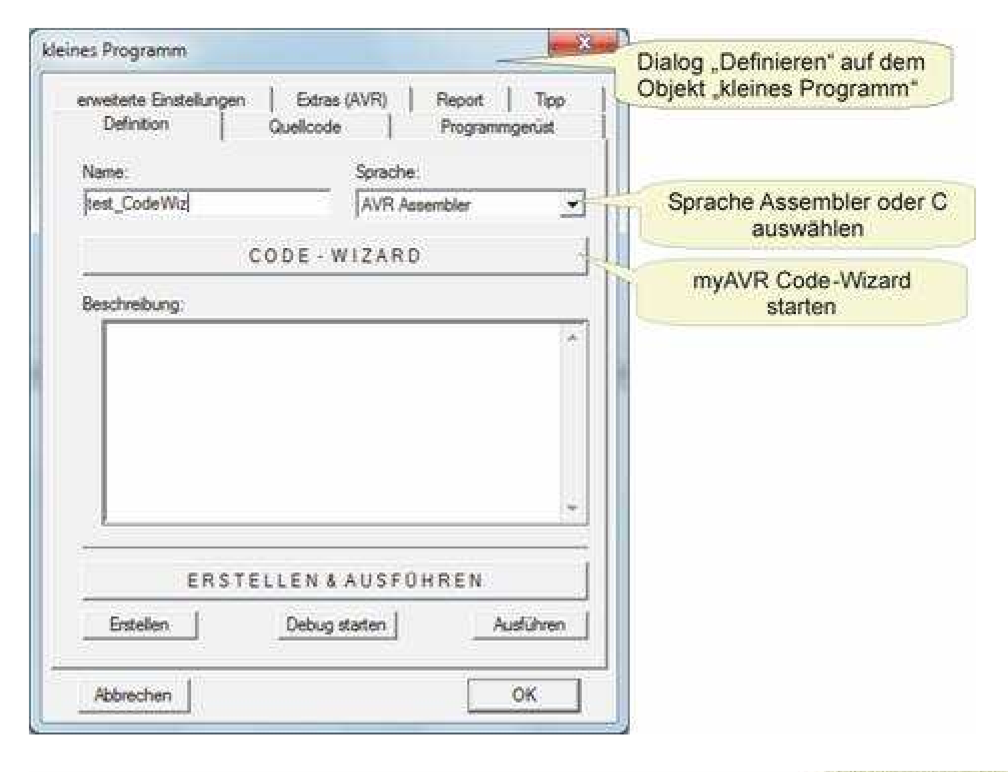
Ein wesentlicher Bestandteil zahlreicher Mikrocontrolleranwendungen ist die Kommunikation mit einem Personalcomputer. Der Simulator verfügt über ein Terminal zur seriellen Kommunikation über die UART des ATmega8 und ist dem myAVR Controlcenter nachempfunden. Sie können zwischen Text-, Hex- und Grafikanzeige wählen.



8 Der myAVR Code-Wizard

8.1 Einführung

Der myAVR Code-Wizard ist ein Assistent zum Erstellen von Assembler- und C-Codes für die Konfiguration und Anwendungsentwicklung von AVR Mikrocontrollern. Dabei wählt der Nutzer Schritt für Schritt im Dialog mit dem Assistenten die gewünschten Konfigurationen und Programmbausteine aus. Der Code-Wizard generiert kompilierungsfähigen Quellcode in der gewünschten Programmiersprache (AVR C oder AVR Assembler), der als komplette Anwendung geladen wird. Der Entwickler muss nur noch die projektspezifische individuelle Logik ergänzen. Der gesamte Programmrahmen in Form von Hauptprogramm, fertige Initialisierungssequenz, Unterprogramme und Interruptroutinen wird von Code-Wizard generiert.





8.2 Grundeinstellungen

Die Zielsprache wurde vor dem Start des Code-Wizards ausgewählt. Für die Generierung von korrektem Quellcode ist es notwendig, den Controllertyp und die Taktgeschwindigkeit festzulegen.



8.3 Geräteeinstellungen

Über den Dialogbereich können Schritt für Schritt die Komponenten des gewählten Controllertyps konfiguriert werden.



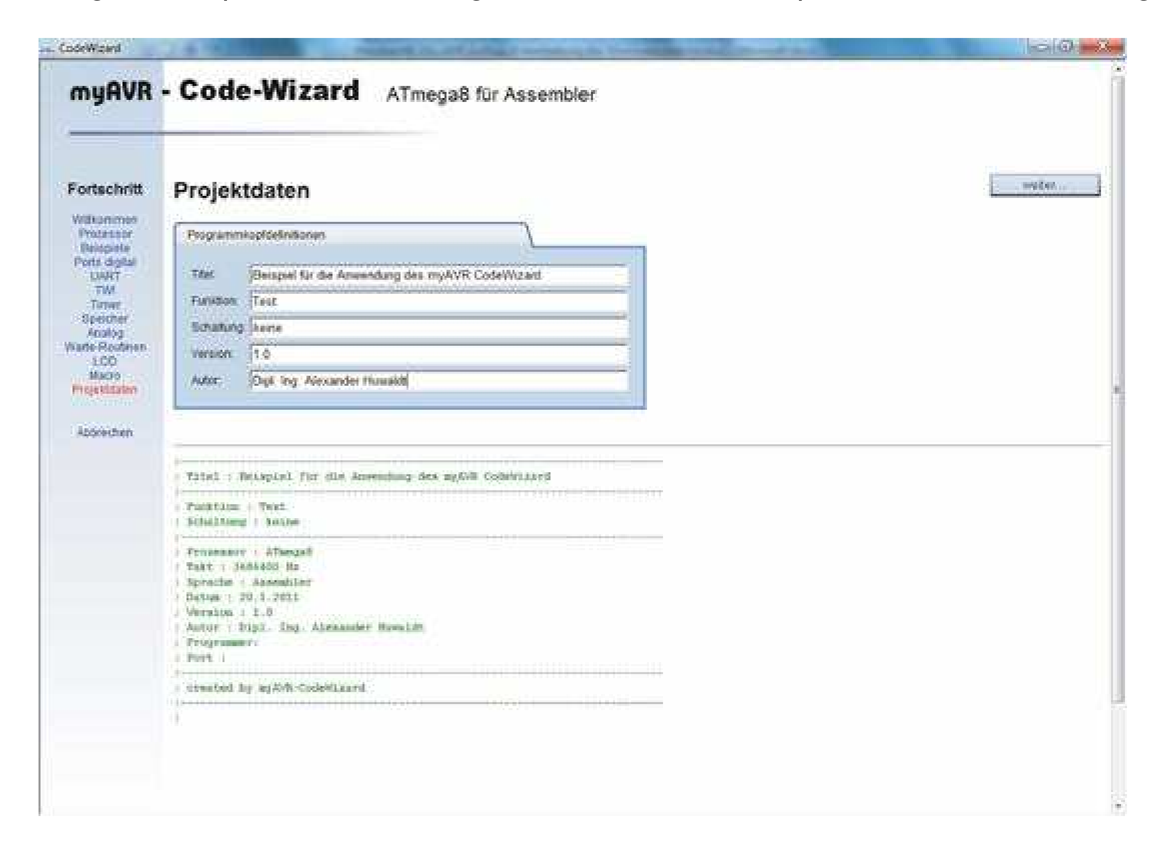
8.4 Unterprogramme

Neben der Möglichkeit die Hardwarekomponenten des gewünschten Controllers zu konfigurieren, bietet der Code-Wizard auch eine Reihe typischer Unterprogramme. So verfügt er über einen Warteroutinen-Rechner zum generieren präziser Wartefunktionen.



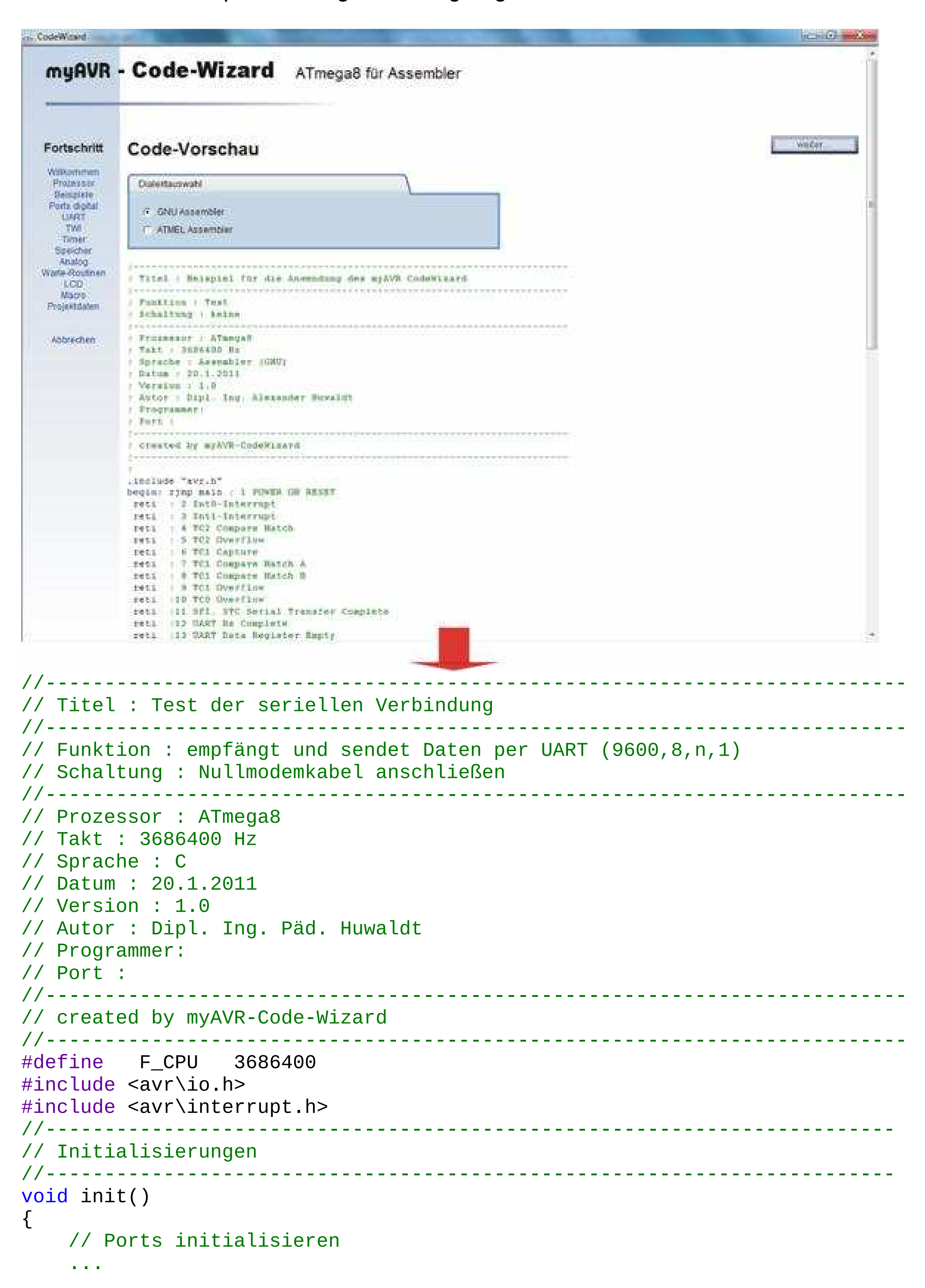
8.5 Projektdaten

Der letzte Punkt im Code-Wizard ist die Eingabe der Projektdaten. Aus diesen wird eine Programmkopfdokumentation generiert und dem kompletten Quellcode vorangestellt.



8.6 Codegenerierung

Der vollständige Quellcode wird zur Kontrolle dem Anwender angezeigt. Es können jetzt noch Änderungen vorgenommen werden, indem man die betreffenden Punkte im Navigationsbereich auswählt und die Parameter ändert. Mit Bestätigung des Quellcodes wird dieser als komplettes Programm eingefügt.




```
//----------------------------------------------------------------------
// Titel : Test der seriellen Verbindung
//----------------------------------------------------------------------
// Funktion : empfängt und sendet Daten per UART (9600,8,n,1)
// Schaltung : Nullmodemkabel anschließen
//----------------------------------------------------------------------
// Prozessor : ATmega8
// Takt : 3686400 Hz
// Sprache : C
// Datum : 20.1.2011
// Version : 1.0
// Autor : Dipl. Ing. Päd. Huwaldt
// Programmer:
// Port :
//----------------------------------------------------------------------
// created by myAVR-Code-Wizard
//----------------------------------------------------------------------
#define   F_CPU   3686400
#include <avr\io.h>
#include <avr\interrupt.h>
//---------------------------------------------------------------------
// Initialisierungen
//---------------------------------------------------------------------
void init()
{
    // Ports initialisieren
    ...
```

9 Entwicklung eines großen Programms mit SiSy AVR

9.1 Einleitung

Für die Entwicklung eines größeren Programms ist es unzweckmäßig, alle Befehle in eine Datei (Unit) zu schreiben. Der Grund dafür ist, dass bei mehr als 60 bis 80 Zeilen Quellcode die Übersicht über die Programmstruktur verloren geht. Selbst die Unterteilung in Unterprogramme reicht ab einer bestimmten Größe von Programmen nicht mehr aus. SiSy erlaubt zwar in kleinen Programmen bzw. je Unit 10 Kilobyte Code. Das sind in Assembler zum Beispiel über 1000 Zeilen. Ein kleines Programm bzw. eine einzelne Unit sollte jedoch nie mehr als 80 bis 120 Zeilen haben. Wird diese Grenze erreicht, sollte das Programm in logische Einheiten (Units, Module) gegliedert werden. Dabei fasst man alles zusammen, was zu einer bestimmten Funktionalität oder einer bestimmten Baugruppe wie zum Beispiel dem AD-Wandler gehört. Physisch entstehen dabei mehrere Dateien, die für sich genommen wieder übersichtlich sind da diese dann nur 80 bis 120 Zeilen Code enthalten. Das Übersetzungsprogramm (Assembler, Compiler, Linker) sorgt dann dafür, dass alle einzelnen Units zu einem vollständigen Programm zusammengefügt werden.

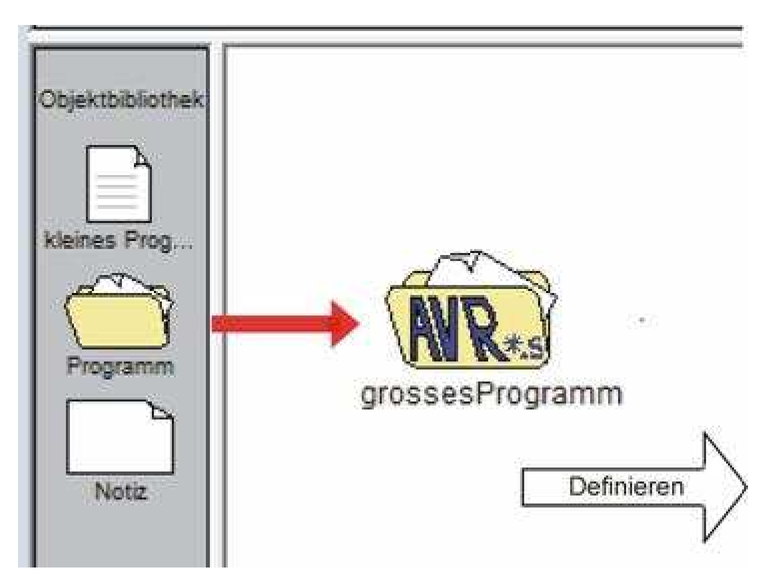
Das folgende Kapitel erläutert an einem sehr kleinen Beispiel die Handhabung der Komponenten eines großen Programms, welches in mehrere Units zerlegt wird.

9.2 Vorbereitung

Starten Sie SiSy und legen Sie ein neues Projekt an. Wählen Sie das Vorgehensmodell „Programmierung“. Nehmen Sie die Grundeinstellungen für die verwendete AVR Hardware vor oder lassen Sie die myAVR-Hardware automatisch suchen. Erstellen Sie ein leeres Diagramm!



Ziehen Sie aus der Objektbibliothek ein Objekt vom Typ „Programm“.





Legen Sie Name und Sprache für das Programm fest. Überprüfen Sie gegebenenfalls die Einstellungen unter Extras AVR.

9.3 Aufgabenstellung

Es ist eine Mikrocontroller-Anwendung mit der Technik des großen Programms (Zerlegen in mehrere Units) zu entwerfen und in der Sprache Assembler zu realisieren.

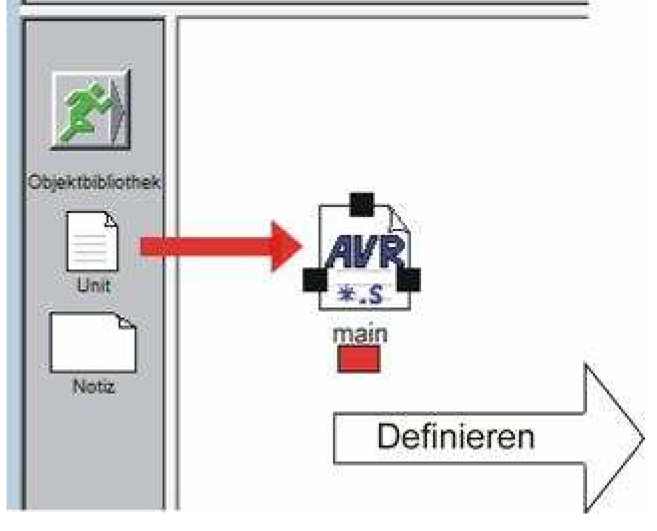
Aufgabe:
Entwickeln Sie eine Mikrocontrollerlösung, bei der ein Taster eine LED schaltet.

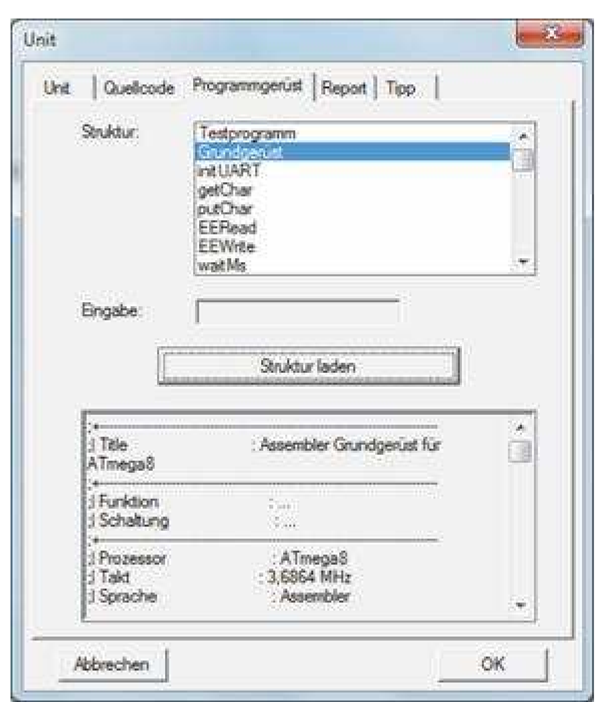
Schaltung:
Port D.2 = Taster 1
Port B.0 = LED

9.4 Hauptprogramm erstellen

Öffnen Sie das Diagrammfenster für ein großes Programm, indem Sie auf dem Symbol rechte Maustaste -> Kontextmenü -> nach unten (öffnen) wählen. Legen Sie eine Unit an. Diese Unit bildet das Hauptprogramm. Nennen Sie die Unit „main". Damit wird durch die Entwicklungsumgebung erkannt, dass es sich hierbei um das Hauptmodul handelt. Erstellen Sie hier das Hauptprogramm, nutzen Sie die angebotene Vorlage „Grundgerüst".



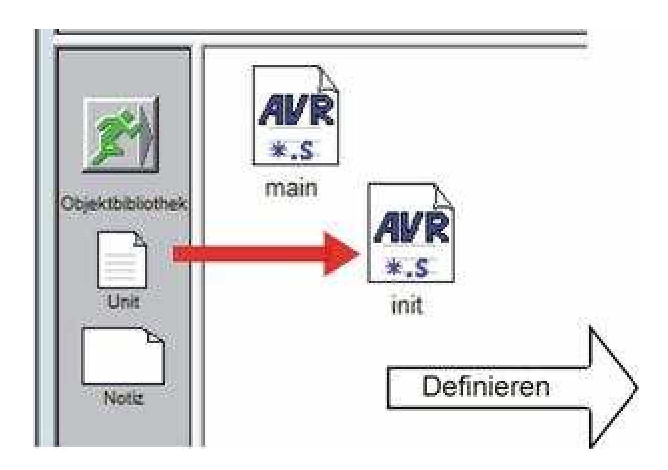


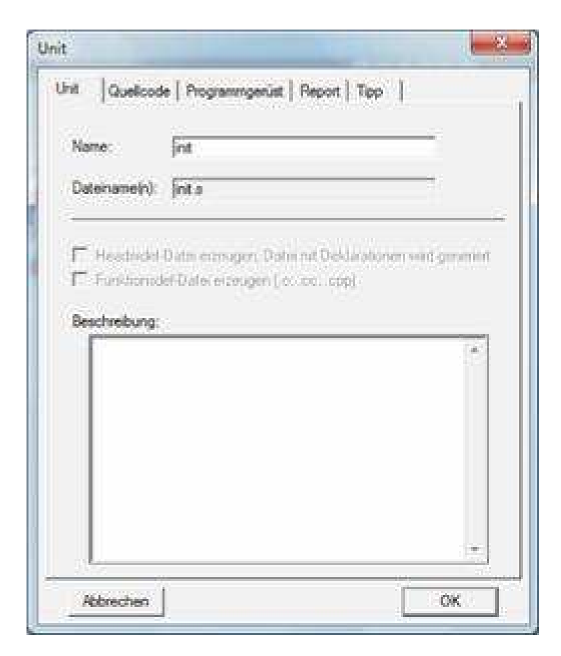



```
;------------------------------------
.include "AVR.H"
;------------------------------------
; Reset and Interrupt vectoren
begin:   rjmp    main
         reti
         reti
         reti
         reti
;...
main:    ldi     r16, lo8(RAMEND)
         out     SPL, r16
         ldi     r16, hi8(RAMEND)
         out     SPH, r16
         ; Hier Init-Code eintragen.
;------------------------------------
mainloop: wdr
          ; Hier Quellcode eintragen.
          rjmp mainloop
```

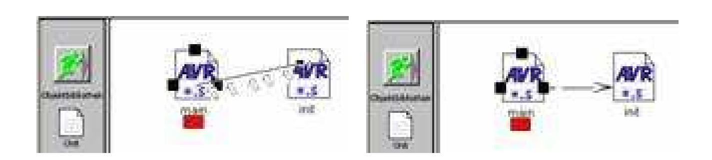
9.5 Units (Unterprogramme) anlegen und verknüpfen

Zur Gliederung des großen Programms wird dieses in mehrere kleine Einheiten (Units/ Module) zerlegt. Diese Einheiten werden nach fachlichen/inhaltlichen Gesichtspunkten gebildet. So kann man alle Initialisierungsaufgaben in der Unit „init“ zusammenfassen. Eine Unit kann aus einer Funktion/Unterprogramm oder mehreren Funktionen/Unterprogrammen bestehen. Im einfachsten Fall enthält jede Unit ein Unterprogramm. Legen Sie zusätzlich zur Haupt-Unit „main“ die Unit „init“ an.



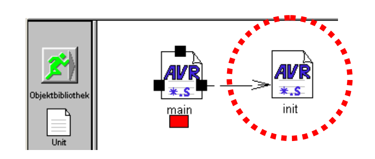


Die Hauptunit benutzt die Unit „init“. Daher ist eine Verbindung von der Hauptunit „main“ zur Unit „init“ zu ziehen. Selektieren Sie die Unit „main“ und ziehen vom Verteiler (rot) eine Verbindung auf die Unit „init“.



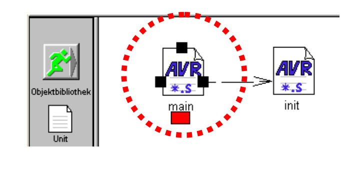
Dabei wird in der Hauptunit „main“ ein Include-Eintrag für die Unit „init“ erzeugt. Erstellen Sie die Initialisierungsroutine für die benötigten digitalen Ein und Ausgänge.

```
;----------------------------------------
init:       push    r16
            sbi     DDRB,0    ; B.0 Ausgang
            cbi     PORTB,0   ; B.0 LED aus
            cbi     DDRD,2    ; D.2 Eingang
            sbi     PORTD,2   ; D.2 PullUp
            pop     r16
            ret
;----------------------------------------
```



Ergänzen Sie den Code des Hauptprogramms.

```
          ...
          rcall   init
;------------------------------------
mainloop: sbic    PIND,2
          rjmp    led_aus
led_an:   sbi     PORTB,0
          rjmp    mainloop
led_aus:  cbi     PORTB,0
          rjmp    mainloop
;------------------------------------
.include “init.s“
```



9.6 Übersetzen, Brennen und Test

Zum Übersetzen, Brennen und Testen wählen Sie im Aktionsmenü den entsprechenden Menüpunkt. Im Ausgabefenster erscheint das Protokoll der ausgeführten Aktionen. Des Weiteren öffnet sich für kurze Zeit das myAVR ProgTool.



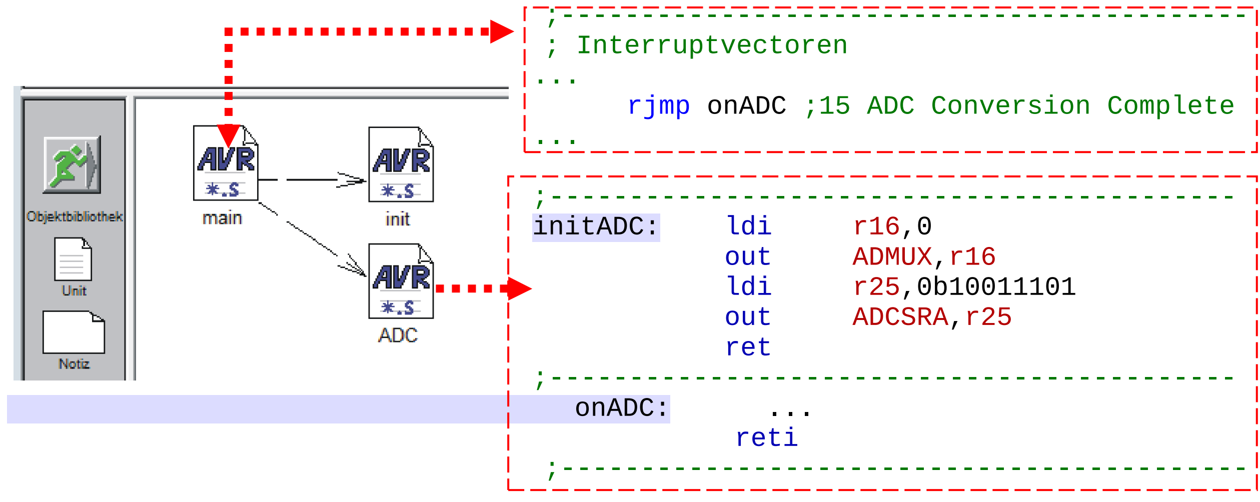
Kompiliere die Datei main.s.
Linke die Datei grossesProgramm.elf.
brenne Daten neu
Öffne myAVR ControlCenter
Ende.

Über das myAVR Controlcenter können Sie das myAVR Board Starten und Stoppen um die Mikrocontrollerlösung zu testen. Überprüfen Sie gegebenenfalls die Einstellungen entsprechend Absatz 6, „Das myAVR Controlcenter".



9.7 Interrupt-Service-Routine (ISR) im großen Programm

Interrupt-Service-Routinen (im weiteren ISR) sind besondere Formen von Unterprogrammen. Diese werden von einer Interruptquelle des Mikrocontrollers (Timer, ADC, UART, usw.) bei entsprechenden Ereignissen automatisch an beliebiger Stelle im Programmfluss aufgerufen (Unterbrechung, engl. Interrupt). Es ist nötig die Interruptquelle entsprechend zu konfigurieren. Es empfiehlt sich für jedes interruptfähige Gerät eine eigene Unit anzulegen. Diese ist mit der Haupt-Unit zu verbinden.

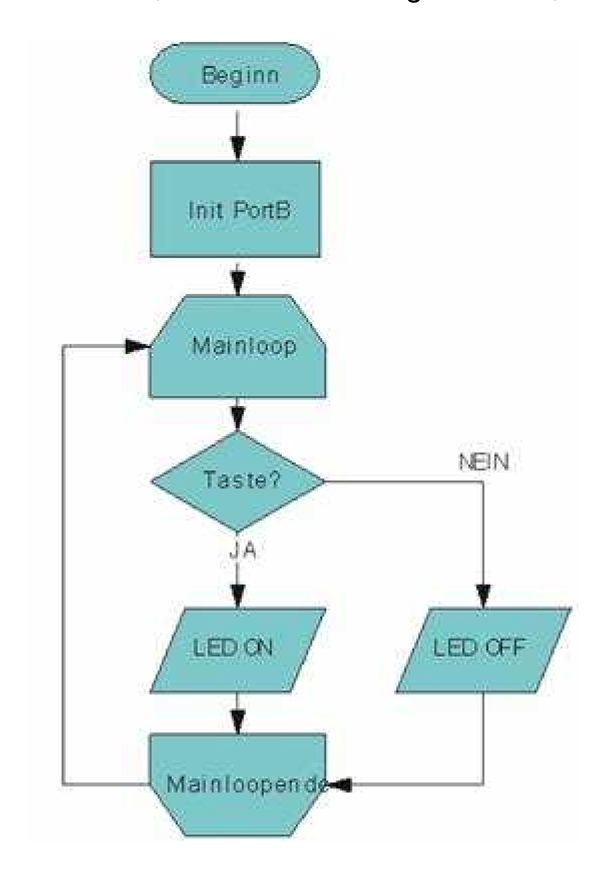


10 Entwicklung eines Programmablaufplans mit SiSy AVR

10.1 Einleitung

Für die Entwicklung eines Programmablaufplans (PAP) sind konkrete Vorstellungen über die Systemlösung und Kenntnis der Hardware nötig. Ein Programmablaufplan kann aus einer genauen Aufgabenstellung abgeleitet werden.

Beispielaufgabe:
Entwickeln Sie eine Mikrocontrollerlösung, bei der ein Taster eine LED schaltet. Der Controller ist so zu initialisieren, dass an Port B.0 der Taster und an Port B.1 die LED angeschlossen ist. Danach ist fortlaufend der Taster abzufragen. Wenn der Taster gedrückt ist, wird die LED eingeschaltet, sonst bleibt die LED aus.

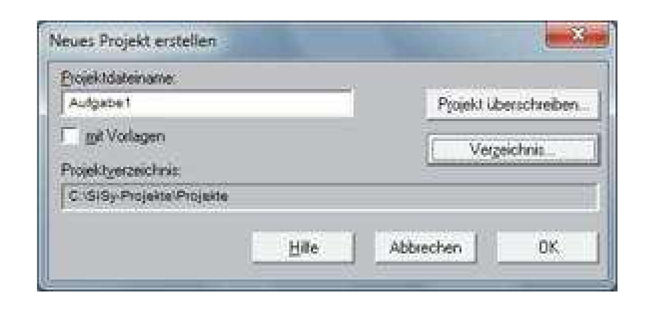


PAP zur Beispielaufgabe

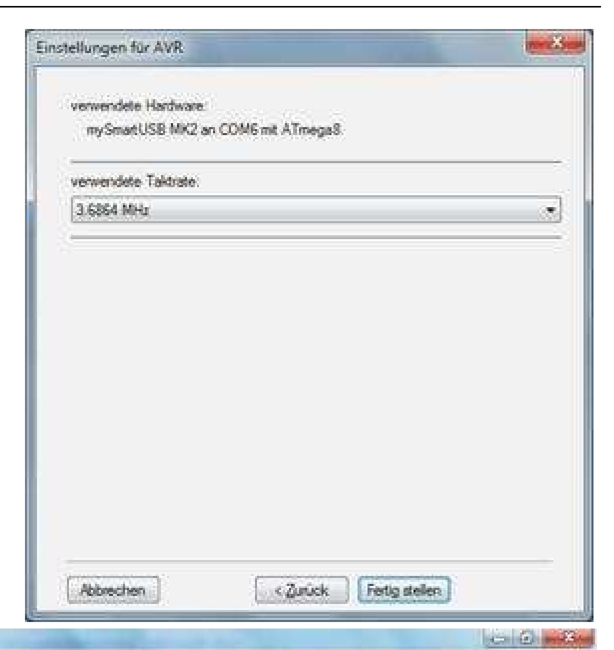
10.2 Vorbereitung

Starten Sie SiSy und legen Sie ein neues Projekt an. Wählen Sie das AVR-Vorgehensmodell. Nehmen Sie die Grundeinstellungen für die verwendete AVR Hardware vor oder lassen Sie die myAVR-Hardware automatisch suchen.



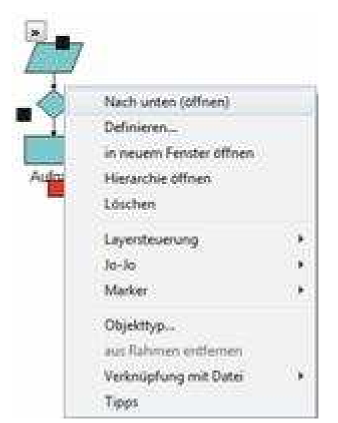


Danach öffnet sich die typische Benutzeroberfläche von SiSy mit einem leeren Vorgehensmodell und Sie können mit der Arbeit beginnen.



Ziehen Sie als nächstes aus der Objektbibliothek ein Objekt vom Typ „PAP“ in das leere Diagramm. Benennen Sie den PAP mit „Aufgabe1“. Beachten Sie die Einstellungen zum Controllertyp und Programmieradapter unter „Extras (AVR)“; vgl. Abbildung, „Anlegen des Objektes „PAP“ und Einstellungen“.

Anlegen des Objektes „PAP“ und Einstellungen



PAP öffnen

Der nächste Schritt ist das Aufstellen des Programmablaufplanes. Dazu muss das Diagramm unter dem Symbol geöffnet werden. Wählen Sie rechte Maustaste „nach unten (öffnen)“, um in dieses Diagramm zu gelangen.

10.3 Aufgabenstellung

Es ist im ersten Schritt eine einfache Mikrocontroller-Anwendung mit der Technik des Programmablaufplanes zu entwerfen und in der Sprache Assembler zu realisieren.

Aufgabe:
Entwickeln Sie eine Mikrocontrollerlösung, bei der ein Taster eine LED schaltet.

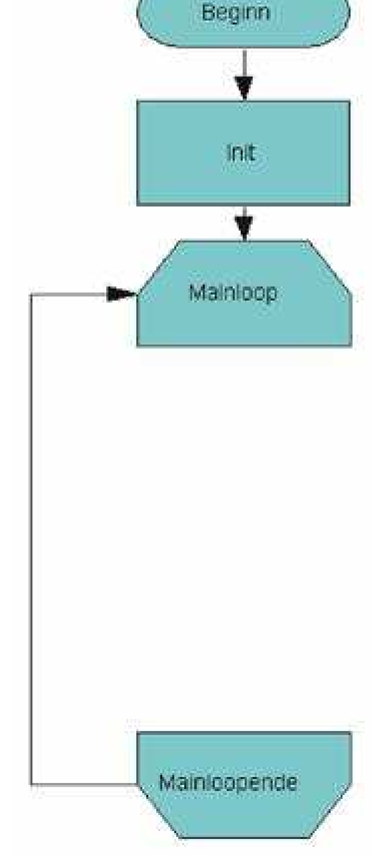
Schaltung:
Port B.0 = Taster 1
Port B.1 = LED

10.4 Grundstruktur laden

Wenn ein Diagramm leer ist, bietet SiSy typische Vorlagen zum Importieren an, siehe Abbildung *„Diagrammvorlagen“*. Diese können dann weiterbearbeitet werden. Wählen Sie die Diagrammvorlage „Mainprogramm-Grundgerüst“.
Abbildung *„Grundgerüst PAP“* zeigt den PAP zu diesem Grundgerüst.



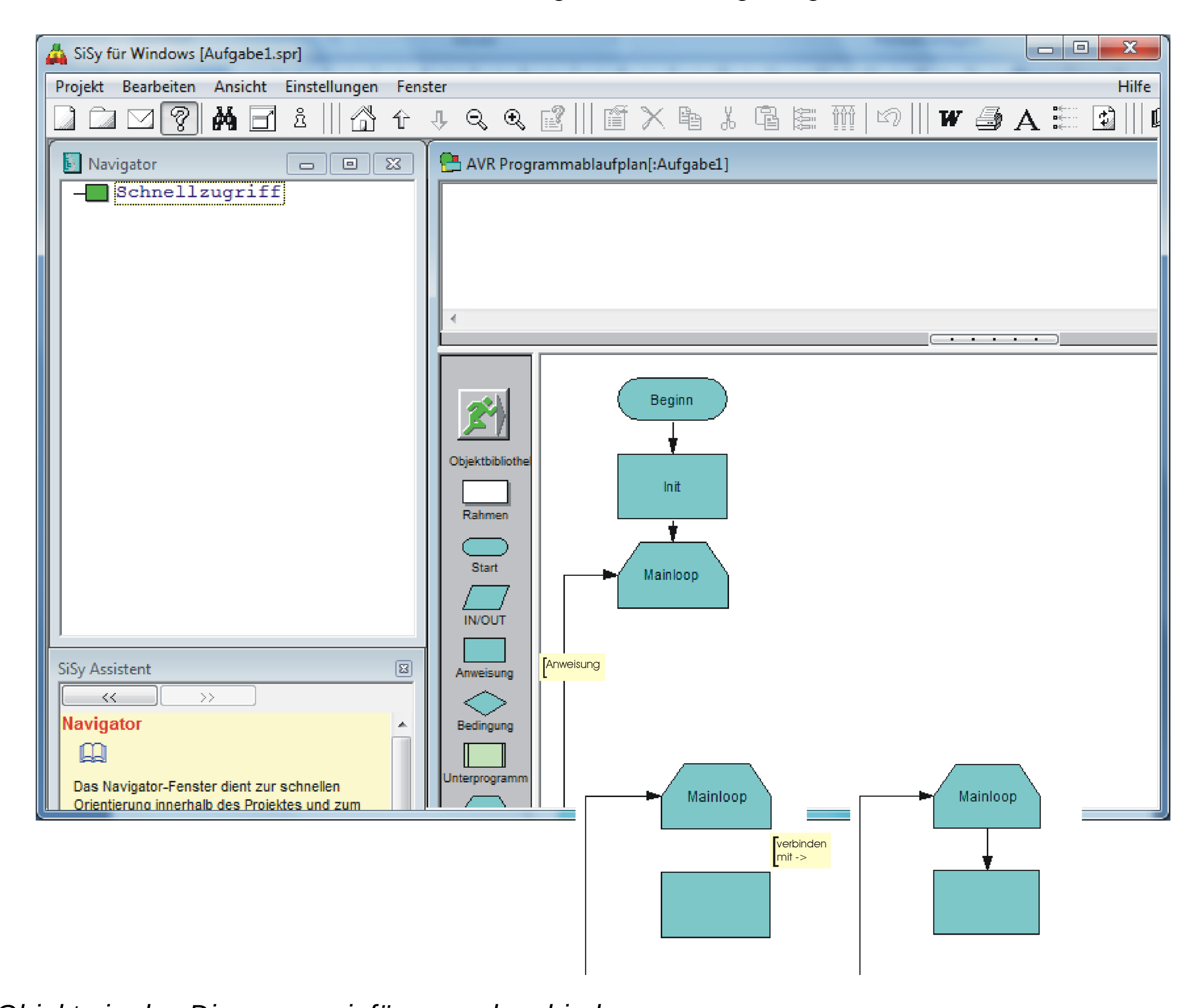
Diagrammvorlagen



Grundgerüst PAP

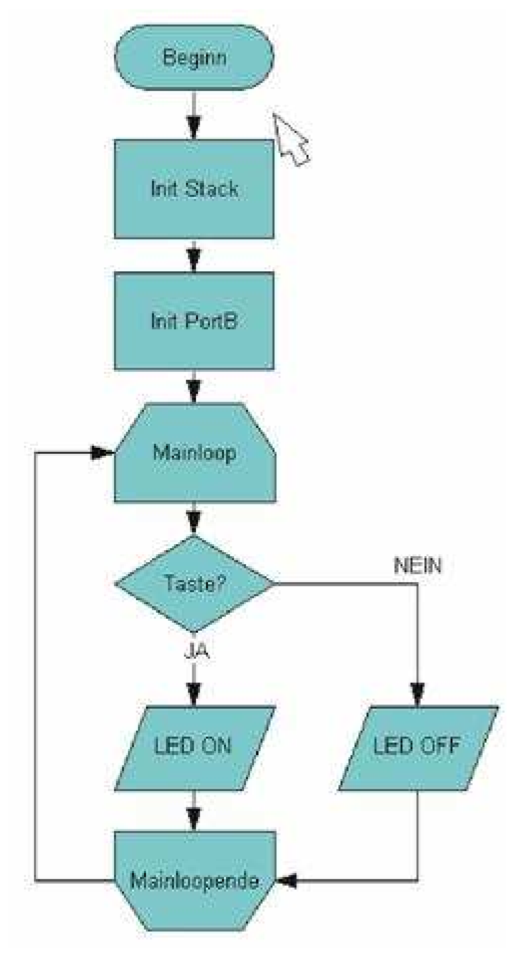
10.5 Logik entwerfen

Um die Programmlogik im PAP abzubilden, muss die Vorlage um die fehlenden Elemente ergänzt werden. Des Weiteren sind die Elemente durch gerichtete Verbindungen (Kanten) in der Reihenfolge ihrer Abarbeitung zu verbinden.
Um ein Objekt im Diagramm zu ergänzen, wird der entsprechende Objekttyp in der Objektbibliothek mit der Maus ausgewählt und per Drag & Drop an die entsprechende Position im Diagramm gezogen. Objekte können mit dem Mauscursor per Klick selektiert und mittels Drag & Drop auch verschoben werden. Selektierte Objekte lassen sich mit der Taste „Entf“ löschen. Verbindungen zwischen den Objekten können über den rot markierten „Verteiler“ von selektierten Objekten hergestellt werden. Dazu ist das Ausgangsobjekt zu selektieren mit dem Mauscursor auf den roten Verteiler zu klicken und bei gedrückter linker Maustaste eine Verbindung zum Zielobjekt zu ziehen. Um Objekte zu benennen, können Sie einen Doppelklick auf dem betreffenden Objekt durchführen oder über rechte Maustaste in den Dialog „Definieren“ gelangen.



Objekte in das Diagramm einfügen und verbinden

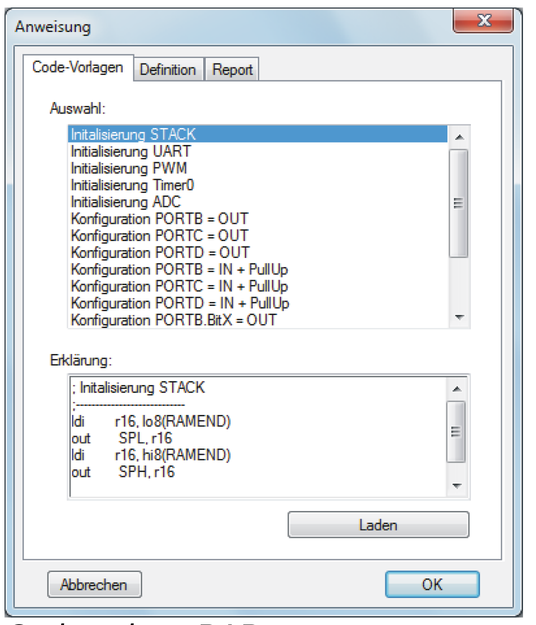
Zeichnen Sie den folgenden Programmablaufplan (vgl. Abbildung „Logikentwurf im PAP“):



Logikentwurf im PAP

10.6 Befehle eingeben

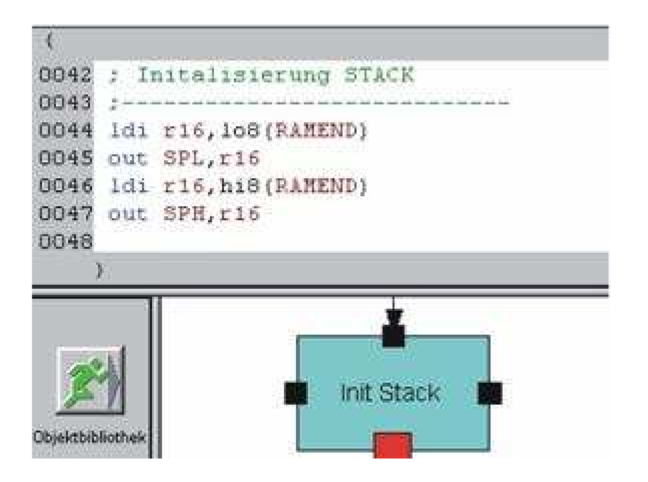
Nachfolgend soll aus dem Programmablaufplan Assemblerquellcode generiert werden. Dazu ist es nötig, die einzelnen Elemente des PAP mit den entsprechenden Assembleranweisungen zu versehen. Dafür gibt es mehrere Möglichkeiten.



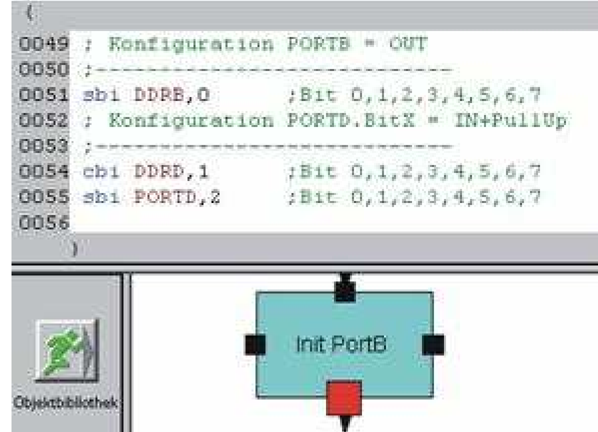
Codevorlage PAP *Definition PAP-Element*

Zum einen bietet SiSy beim ersten Öffnen eines jeden Elementes typische Code-Vorlagen an, die über die Schaltfläche „Laden“ dem Element zugewiesen werden können. Wird der Definieren-Dialog mit der Schaltfläche „OK“ beendet, so wird die Auswahl im Objekt gespeichert und beim nächsten Aufruf des Dialoges „Definieren“ erscheinen die Code-Vorlagen nicht mehr, sondern das Element kann ganz normal bearbeitet werden. In Abbildung *„Codevorlage PAP“* und *„Definition Pap-Element“* sind beide Varianten des Dialoges „Definieren“ zu sehen“.

Die zweite Möglichkeit besteht beim Selektieren von Elementen über den Quellcodeeditor oberhalb des Diagrammfensters, vgl. Abbildung „Quellcodefenster im PAP“ und Abbildung „Quellcodeeingabe PAP“.



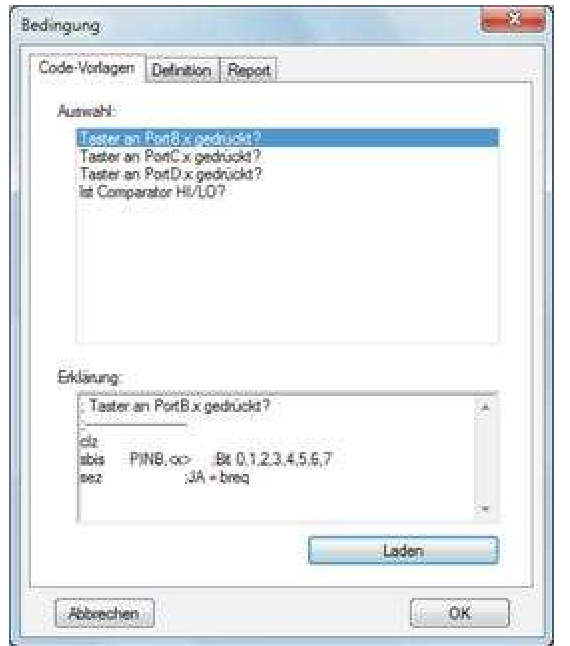
Quellcodefenster im PAP



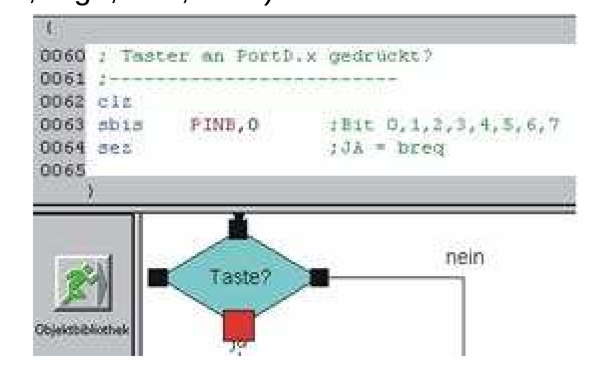
Quellcodeeingabe PAP

Geben Sie die gezeigten Quellcodes in die Objekte ein!

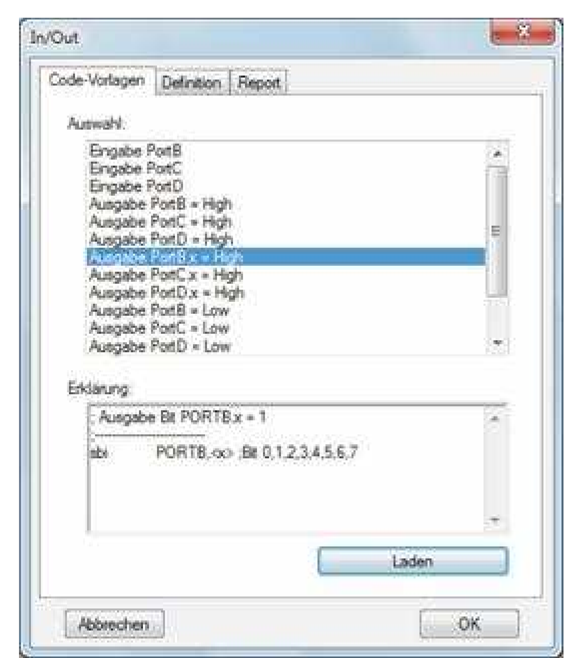
Bedingungen haben spezielle Vorlagen, die eine Codegenerierung bei übersichtlichem Programmablaufplan vereinfachen.
Jede Bedingungsvorlage ist so konstruiert, dass eine JA/NEIN Entscheidung erzeugt werden kann. Findet der Codegenerator das Schlüsselwort JA an einer der folgenden Verbindungen, setzt er diese in eine Sprunganweisung „breq“ um. Das Schlüsselwort NEIN wird in „brne“ umgewandelt. Alternativ können statt dieser Schlüsselworte auch der Sprungbefehl selber an eine der Kanten geschrieben werden (breq, brne, brge, brlo, usw.)



Codevorlage Bedingung



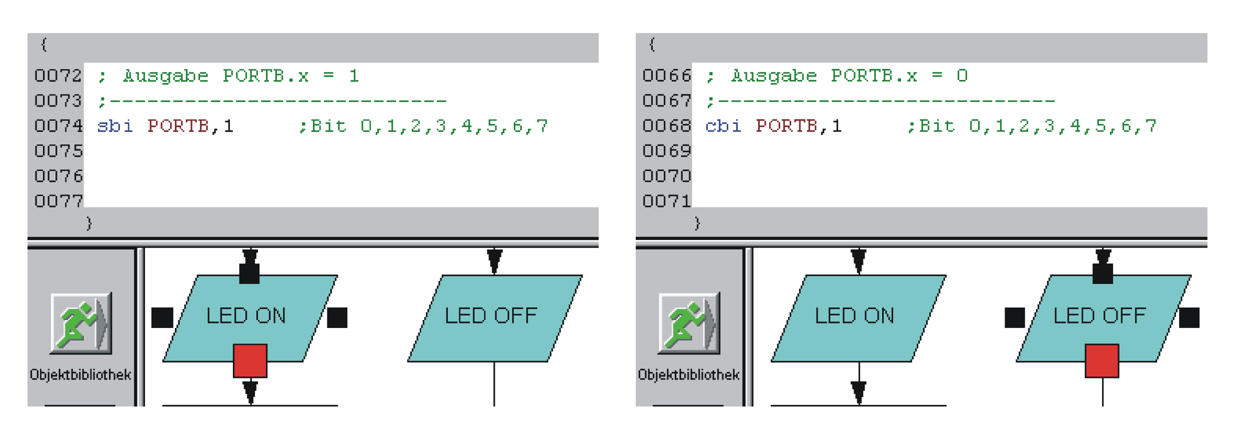
Quellcode einer Bedingung



Vorlagen für IN/OUT

Auch das Element „IN/OUT" verfügt über spezifische Vorlagen. Diese sind gegebenenfalls mit zu ergänzen. Dazu sind spitze Klammern als Platzhalter in den Vorlagen eingefügt.

Ergänzen Sie den Quellcode der gezeigten Elemente!

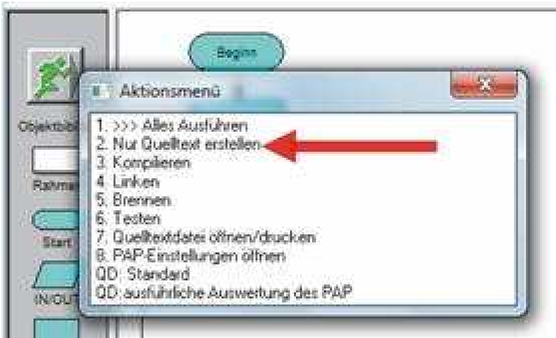


10.7 Übersetzen, Brennen und Test

Sind alle betreffenden Elemente mit Quellcode hinterlegt, kann aus dem Programmablaufplan der komplette Quellcode generiert, kompiliert, gelinkt und auf den Mikrocontroller übertragen werden. Die gewünschte Funktion kann aus dem Aktionsmenü ausgewählt werden (Abbildung *„Auswahl aus Aktionsmenü").*

Hinweis:

Beachten Sie, dass für das Brennen des Controllers das Programmierkabel angeschlossen sein muss und bei Bedarf eine geeignete Spannungsquelle anzuschließen ist.

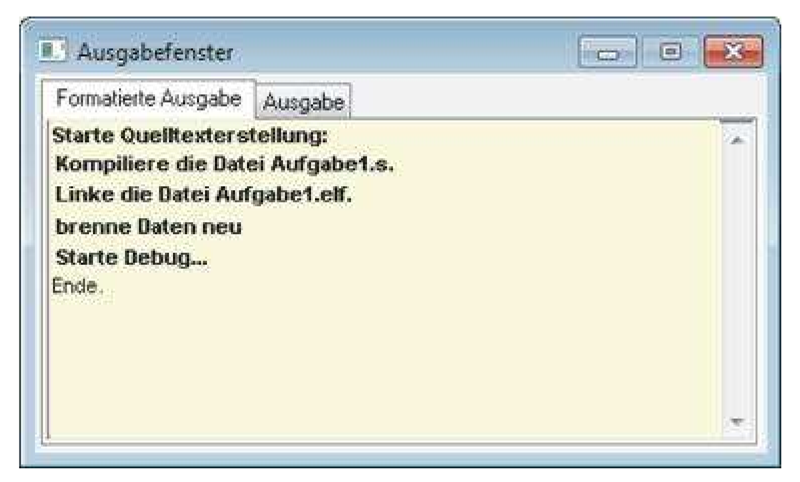


Funktionalitäten des Aktionsmenüs	
Benennung	**Funktion**
Alles Ausführen	Quellcode generieren, kompilieren, linken, brennen
Nur Quelltext erstellen	Quellcode generieren mit allen Marken
Kompilieren	Quellcode zu Objektdatei übersetzen
Linken	Objektdatei zu ELF-Datei binden
Brennen	ELF-Datei an den Controller übertragen

Auswahl aus Aktionsmenü

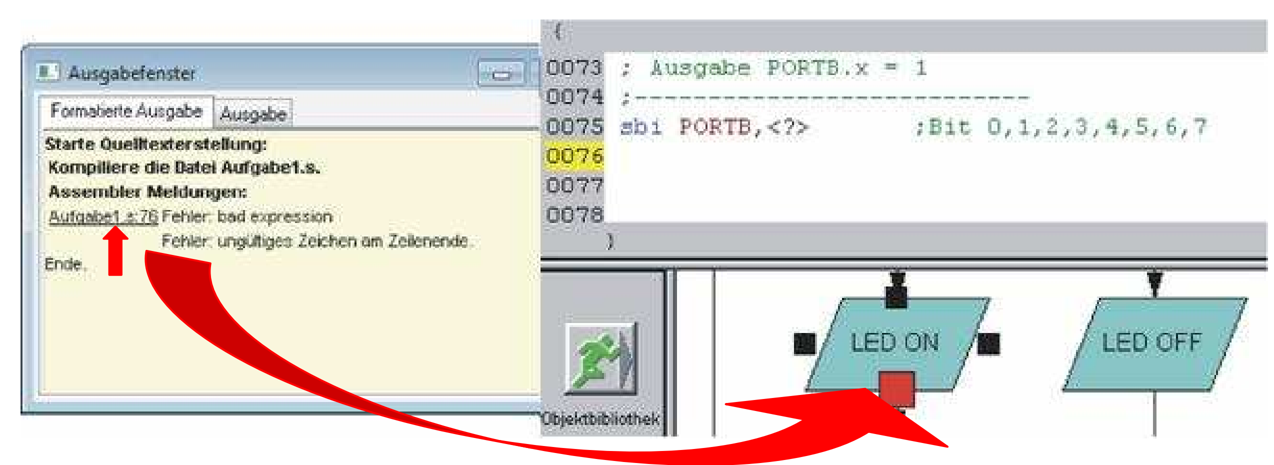
Testen	myAVR Controlcenter öffnen
Quelltextdatei öffnen	Quellcodedatei öffnen
Quelltext bereinigen	Quellcode von überflüssigen Marken bereinigen

Im Ausgabefenster werden die jeweiligen Aktionen angezeigt.

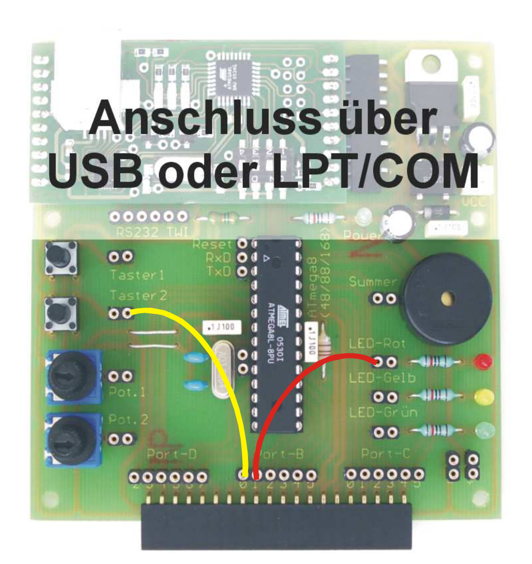


Ausgabefenster

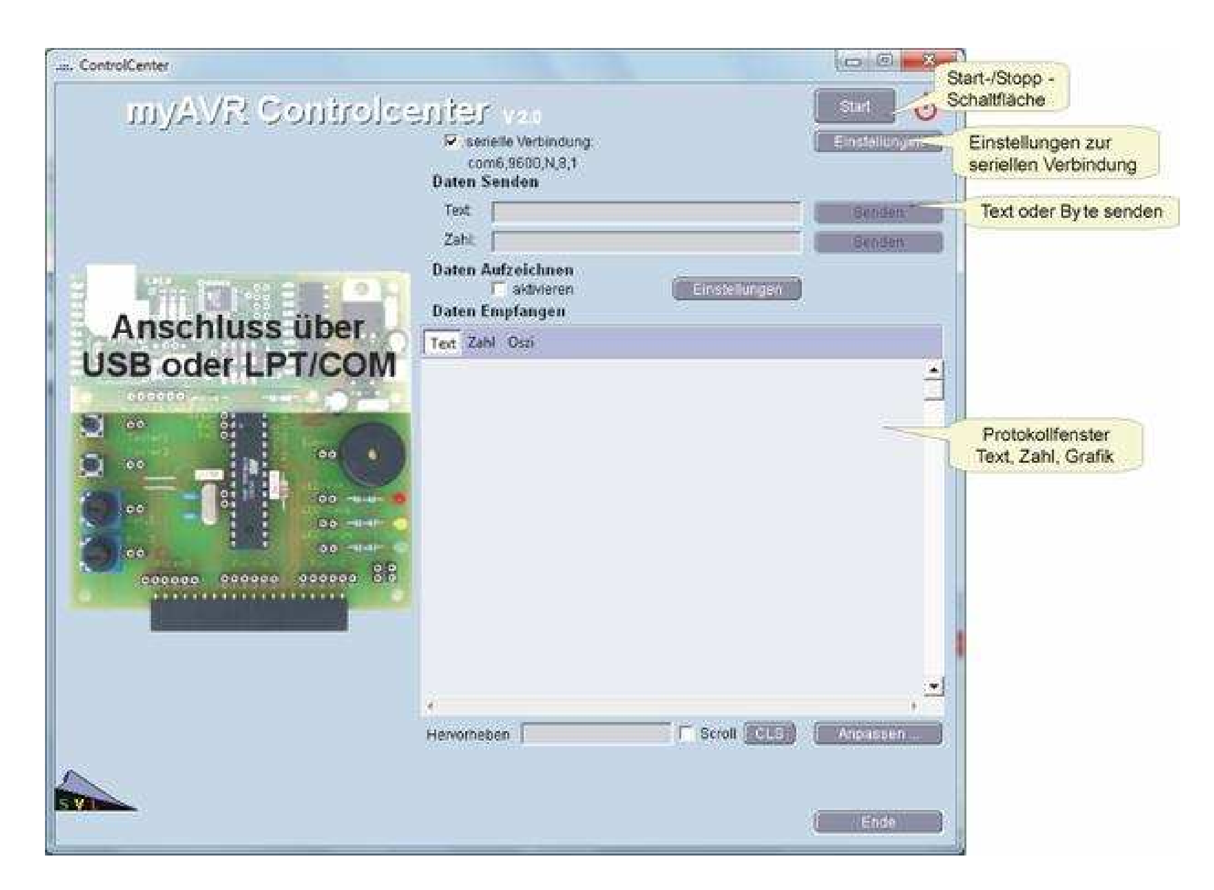
Bei Compilerfehlern werden diese ebenfalls im Ausgabefenster mit der entsprechenden Zeilennummer angezeigt. Um zu dem Fehler zu gelangen, genügt meist ein Klick auf die Fehlermeldung. Das betreffende Objekt wird selektiert und die Zeile hervorgehoben.



Fehlerbehandlung



Nachdem das Programm erfolgreich übersetzt und auf den Controller übertragen wurde, kann die Anwendung getestet werden. Dazu ist bei Bedarf das myAVR Controlcenter zu öffnen und die vorgegebenen Verbindungen auf dem Board zu stecken. Über die Schaltfläche „Start/Stopp“ kann das myAVR Board ein und ausgeschaltet werden.

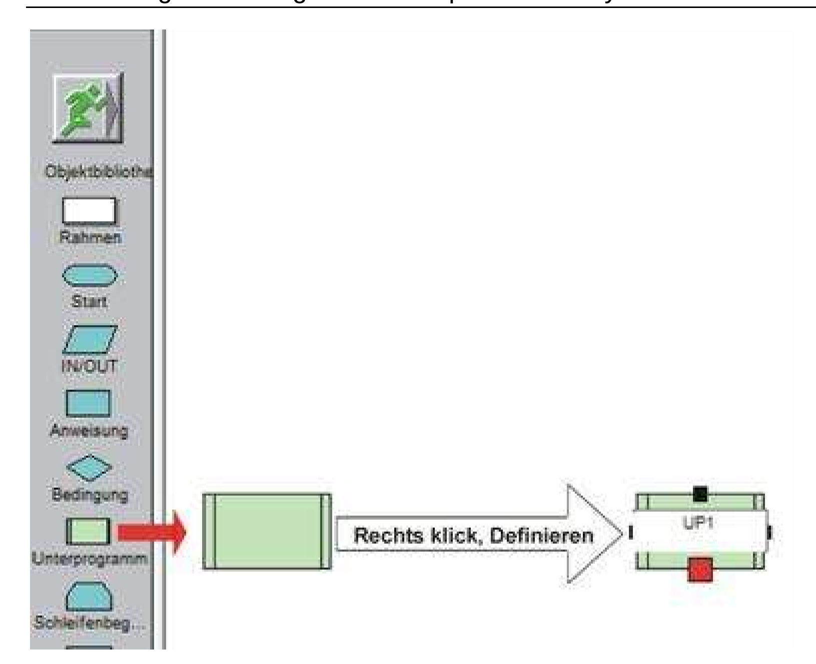


10.8 Unterprogrammtechnik im PAP

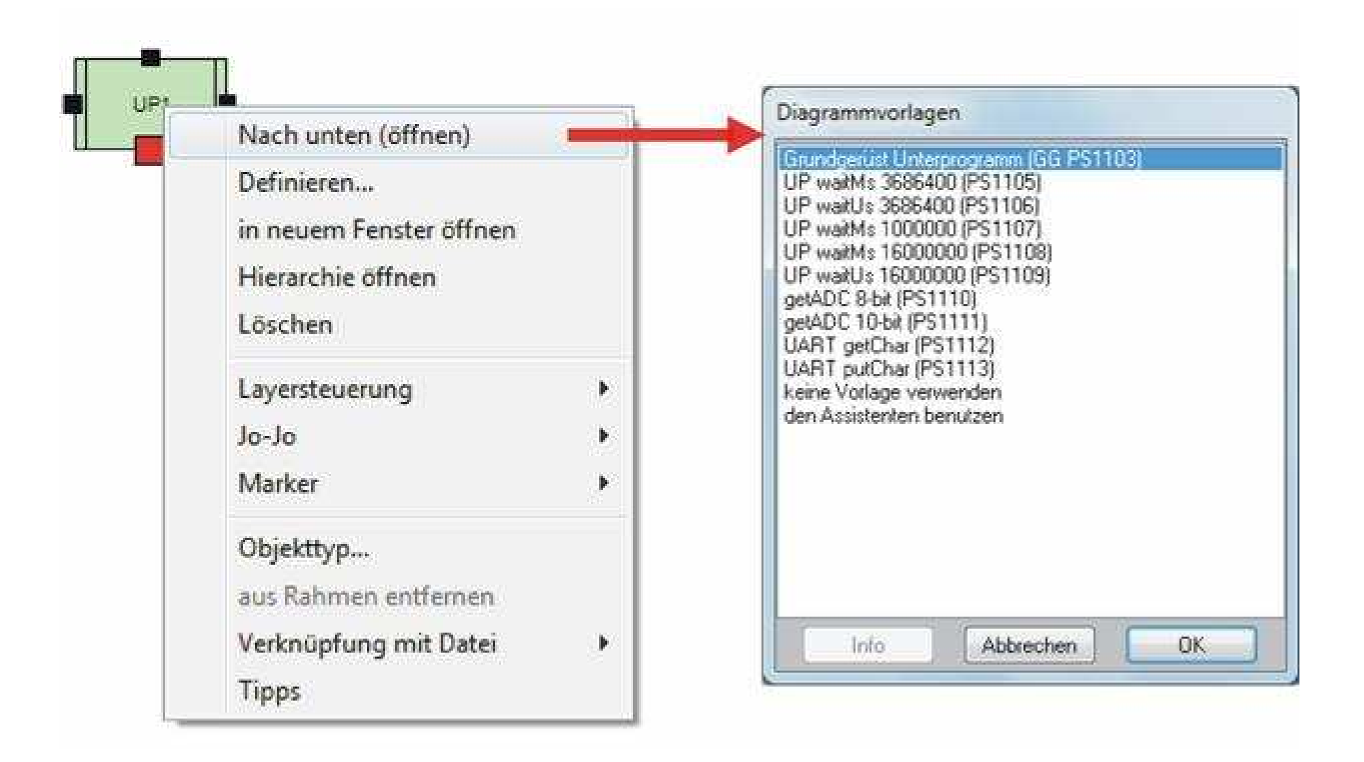
Unterprogramme sind ein wichtiges Gestaltungsmittel für übersichtliche Mikrocontrollerprogramme. Sie werden für in sich abgeschlossene Aufgaben (Verarbeitungsschritte) benutzt, die auch mehrfach im Gesamtprogramm genutzt werden können.

10.8.1 Anlegen eines Unterprogramms

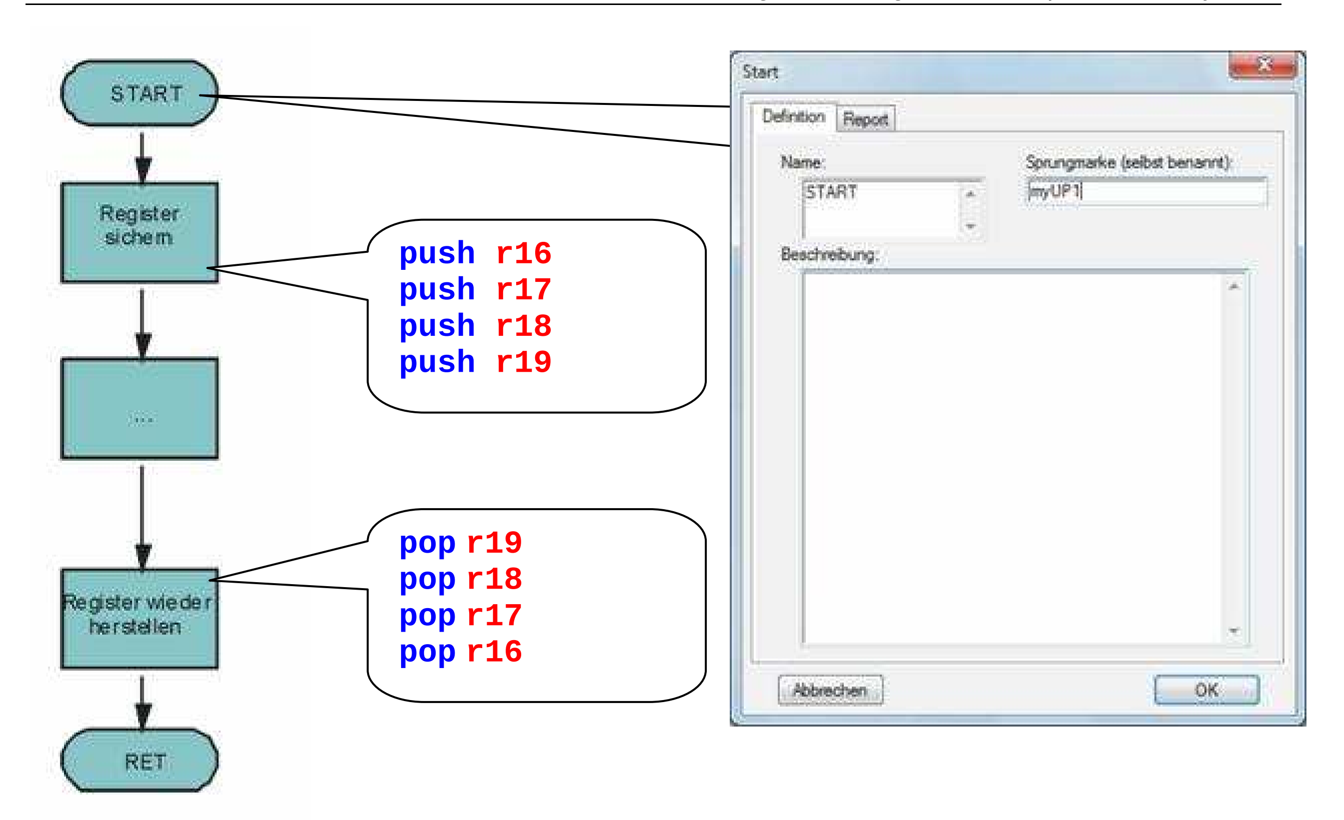
Ziehen Sie den Objekttyp „Unterprogramm“ aus der Objektbibliothek in das gewünschte Diagramm. Mit Doppelklick oder über rechte Maustaste -> Kontextmenü -> Definieren auf dem Element können Sie dem Unterprogramm einen Namen geben.



Damit ist ein Objekt angelegt, welches im aktuellen Diagramm als Aufruf (call) des Unterprogramms zu verstehen ist. Die Funktionalität des Unterprogramms wird in einem gesonderten Programmablaufplan für das Unterprogramm entworfen. Dazu ist das Diagramm „unter“ bzw. „hinter“ Unterprogramm zu öffnen. Um das zum Unterprogramm zugehörige Diagramm zu öffnen, wählen Sie auf dem Objekt rechte Maustaste -> Kontextmenü -> Nach unten (öffnen).

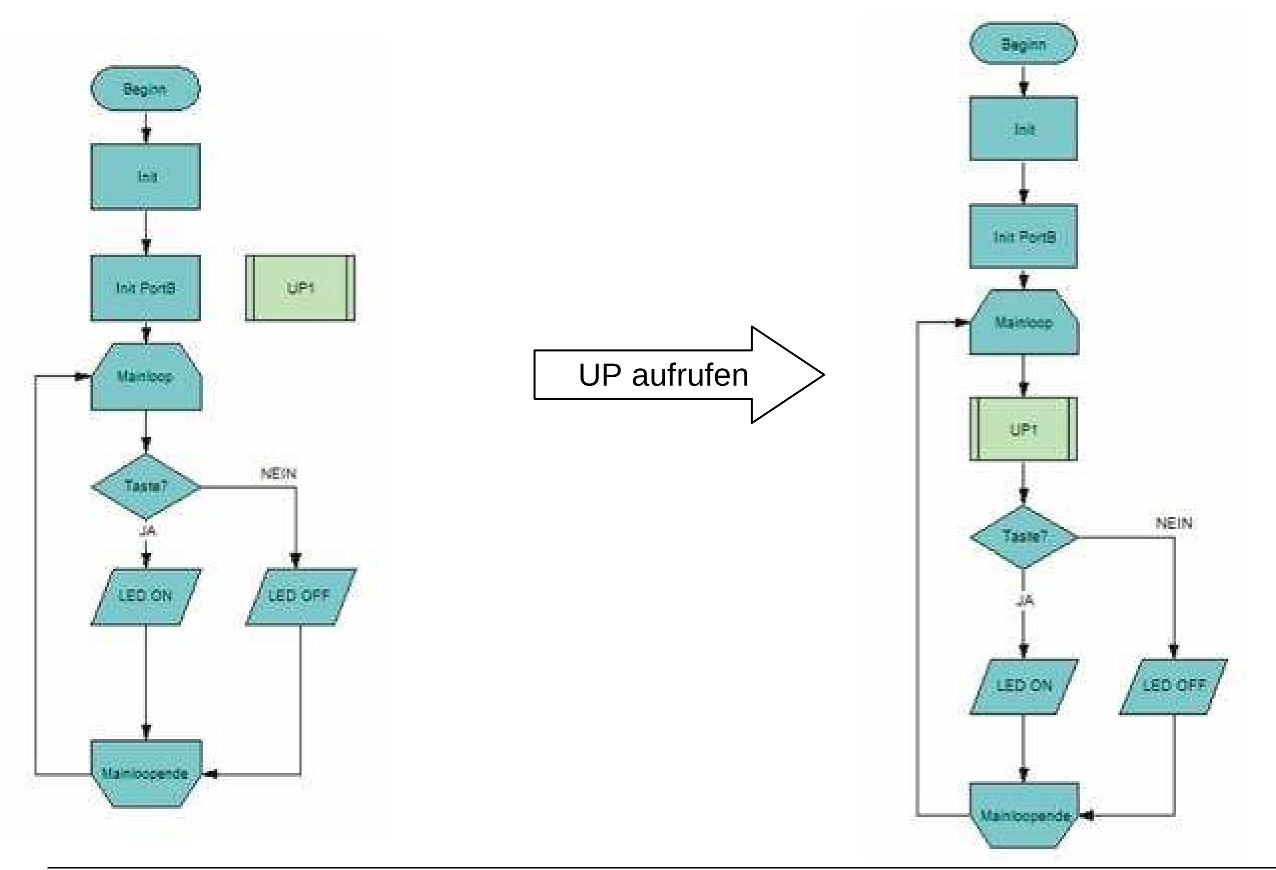


Sie erhalten eine Vorlagenliste für die Grundstruktur von Unterprogrammen. Bitte laden Sie die Vorlage „Grundgerüst Unterprogramm“. Auf dem Objekt „START“ können Sie eine benutzerdefinierte Sprungmarke festlegen (Rechtsklick -> Definieren), die durch den Codegenerator erstellt und verwendet werden soll. Die Vorlage muss entsprechend der vorgesehen Logik abgeändert werden.



10.8.2 Ein Unterprogramm aufrufen

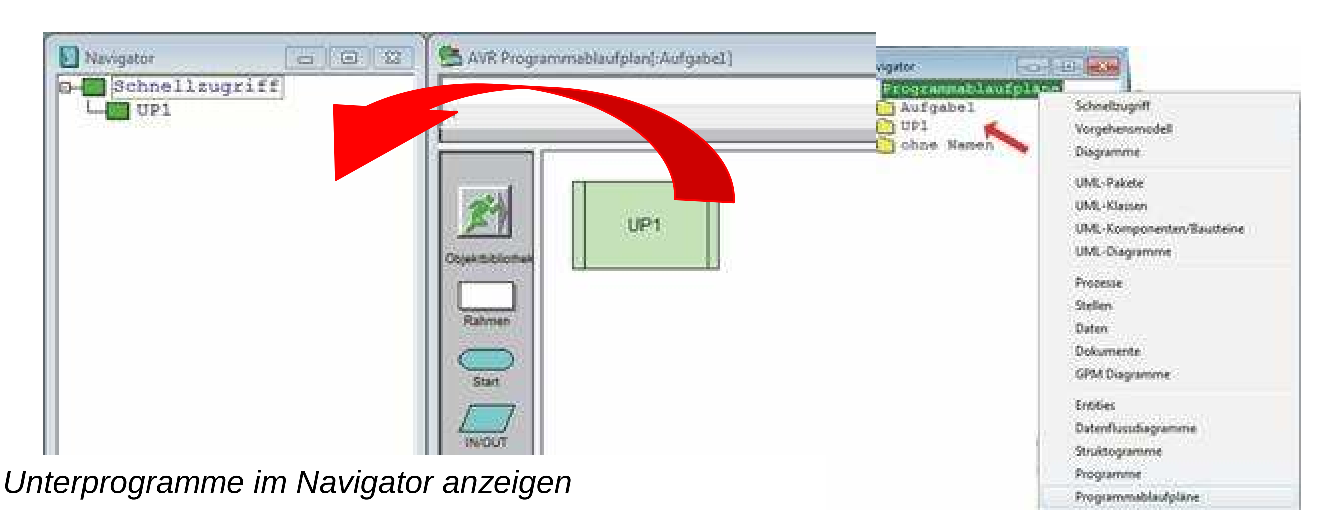
Das Unterprogrammsymbol muss zum Aufruf an der entsprechenden Position im Programmablaufplan eingefügt werden. Der Codegenerator erzeugt dann entsprechend einen Unterprogrammaufruf und den Code für das Unterprogramm selbst. Dazu ist in das Diagramm zurückzukehren, in dem das Symbol „Unterprogramm“ angelegt wurde (zum Beispiel rechte Maustaste -> Kontextmenü-> nach oben…). Das Unterprogramm ist korrekt eingebunden, wenn es vollständig und eindeutig im Programmfluss integriert ist (mindestens ein eingehender Pfeil und genau ein ausgehender Pfeil).



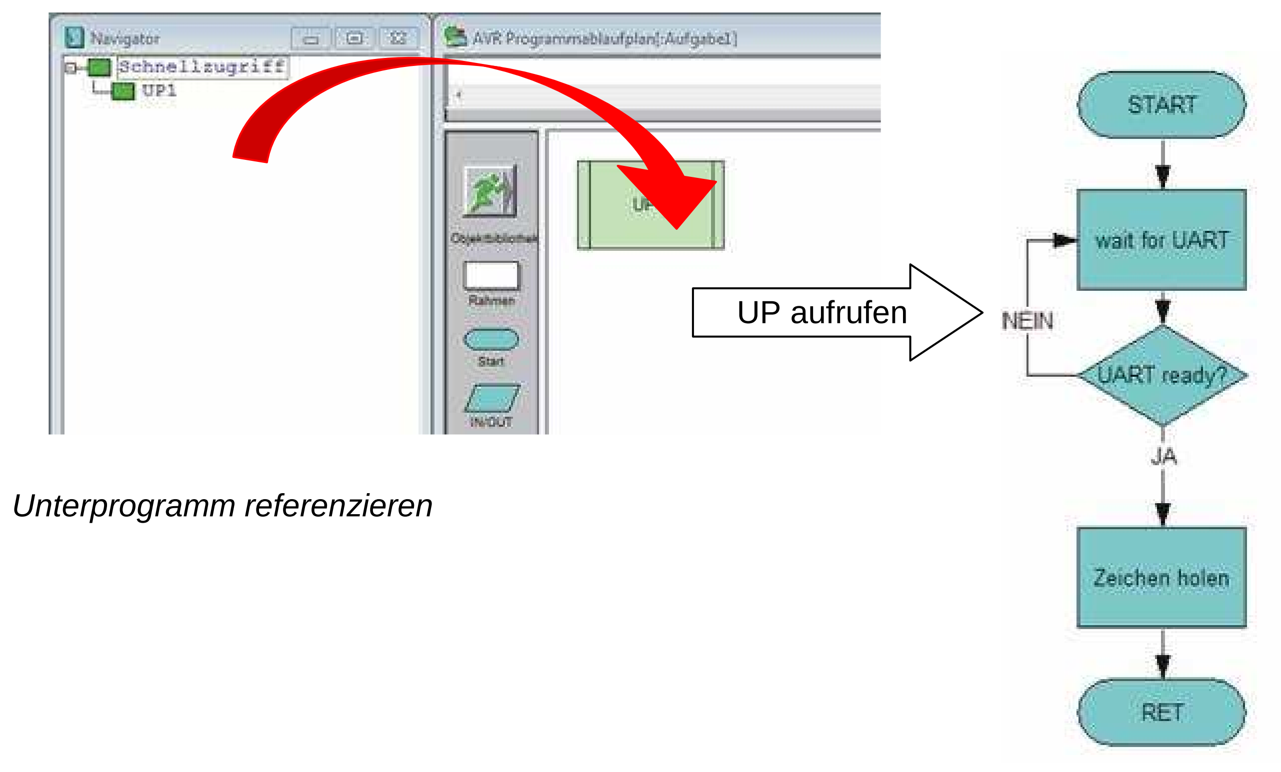
10.8.3 Unterprogramme mehrmals benutzen

Ein wesentliches Merkmal von Unterprogrammen ist, dass diese von verschiedenen Stellen im Programm aufgerufen (call) werden können und auch dorthin zurückkehren (return). Um diese Möglichkeit zu nutzen, bietet SiSy das Anlegen von Referenzen an. Vergleichen Sie dazu Absatz 3.2 „Die Modellierungselemente von SiSy“. Um ein Unterprogramm zu referenzieren (wiederholend zeigen und einbinden) gehen Sie wie folgt vor:

1. zeigen Sie im Navigator das gewünschte Unterprogramm an
    a. über den Schnellzugriff,
       dort lässt sich das Original per Drag & Drop ablegen, oder
    b. über die Navigatorsortierung „Unterprogramme“
       Navigator -> rechte Maustaste -> Kontextmenü -> Programmablaufpläne
2. das gewünschte Zieldiagramm, in dem das Unterprogramm verwendet werden soll, öffnen
3. per Drag und Drop das Unterprogramm in das Zieldiagramm ziehen, dabei wird nur eine Referenz (Link) auf das Original angelegt.
4. Referenz wie oben beschrieben in den Programmfluss integrieren



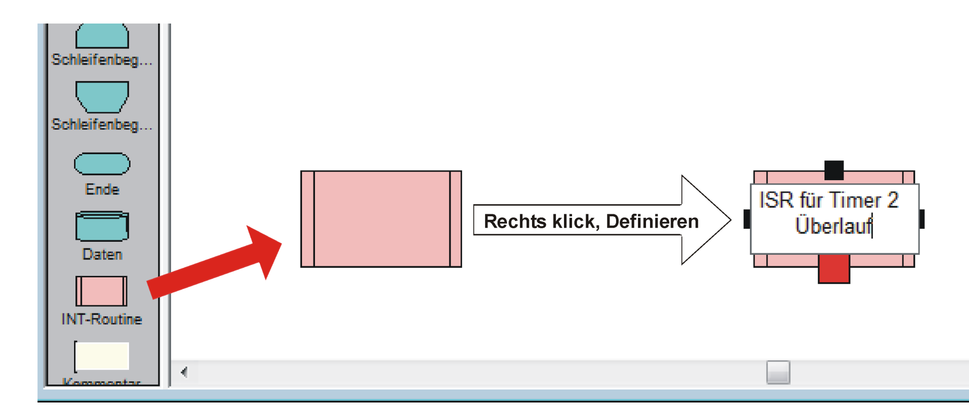
Unterprogramme im Navigator anzeigen



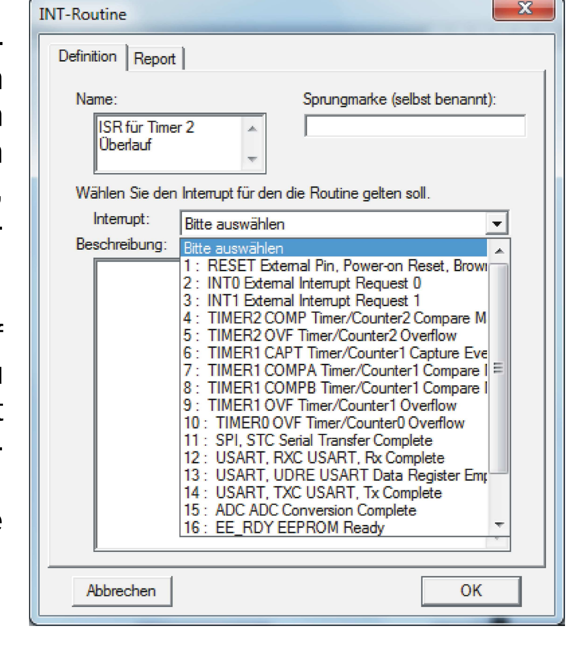
Unterprogramm referenzieren

10.9 Interrupt-Service-Routinen (ISR) im PAP

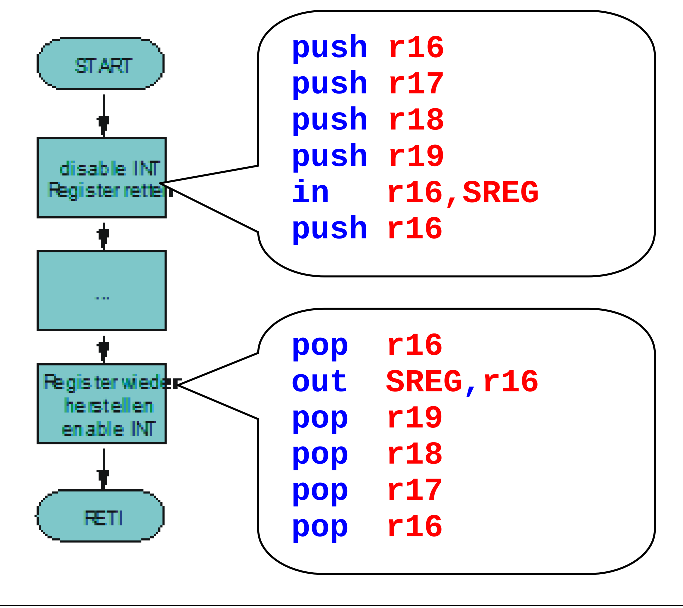
Interrupt-Service-Routinen (im weiteren ISR) sind besondere Formen von Unterprogrammen. Diese werden von einer Interruptquelle des Mikrocontrollers (Timer, ADC, UART, usw.) bei entsprechenden Ereignissen automatisch an beliebiger Stelle im Programmfluss aufgerufen (Unterbrechung, engl. Interrupt). Es ist demzufolge nicht vorgesehen, eine ISR in den Programmfluss zu integrieren.
Um eine ISR zu erzeugen, ziehen Sie ein Objekt vom Typ INT-Routine aus der Objektbibliothek in das Diagramm des Hauptprogramms und definieren einen Namen.

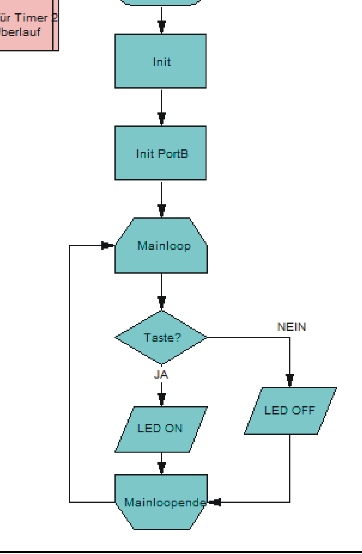


Über rechte Maustaste -> Kontextmenü -> Definieren … legen Sie die Sprungmarke und den Typ des Interrupt fest. Damit erfolgt durch den Codegenerator die Zuordnung der ISR zum entsprechenden Interruptvektor. Beachten Sie, dass die Liste der Interrupts abhängig vom gewählten Controllertyp ist.



Zum Entwerfen der ISR-Logik wählen Sie auf dem Objekt rechte Maustaste -> Kontextmenü -> nach unten … Ihnen wird das Grundgerüst einer ISR als Diagrammvorlage angeboten. Laden Sie die Diagrammvorlage.
Vervollständigen Sie danach die ISR-Logik. Die ISR wird nicht in den Programmfluss integriert.



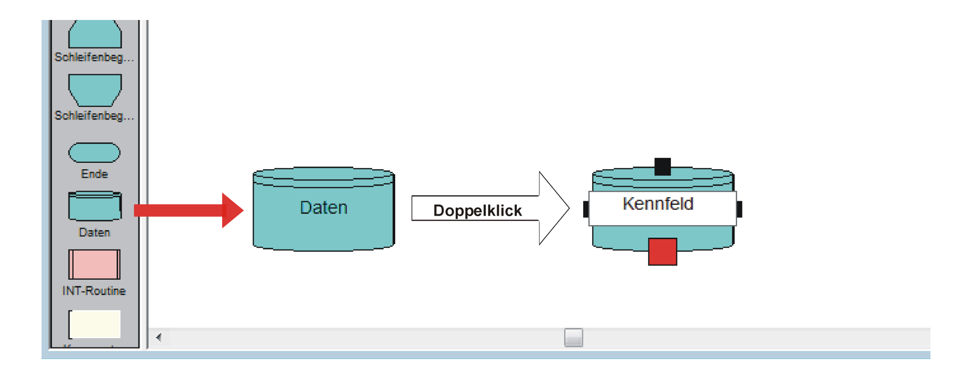


10.10 Daten im PAP

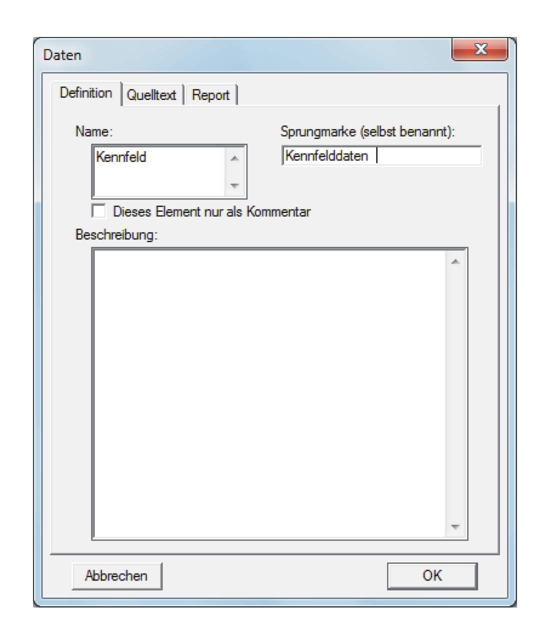
Konstante Daten, die im Programmspeicher (FLASH) des Mikrocontrollers abgelegt werden, können als Datenobjekt deklariert werden. Die im Datenobjekt deklarierten Daten werden durch den Codegenerator immer ans Ende des Quellcodes gestellt. Die zu generierende Marke/Marken für die Datenelemente können vom Entwickler frei gewählt werden.

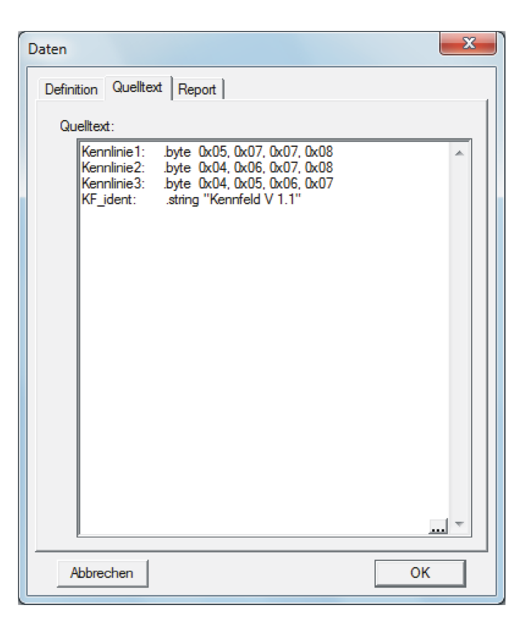
10.10.1 Anlegen eines Datenobjektes

Zum Anlegen eines Datenobjektes ziehen Sie das betreffende Objekt per Drag & Drop aus der Objektbibliothek in das gewünschte Diagramm. Per Doppelklick können Sie einen Namen vergeben (Beachte: Menüfolge *Einstellungen/Menü bei Doppelklick*).



Die Marke und die Daten selbst können über rechte Maustaste -> Kontextmenü -> Definieren… festgelegt werden.





10.10.2 Datenobjekt benutzen

Im Quellcode werden die Daten über die vergebenen Markennamen angesprochen.

AVR Programmablaufplan[:Aufgabe1]

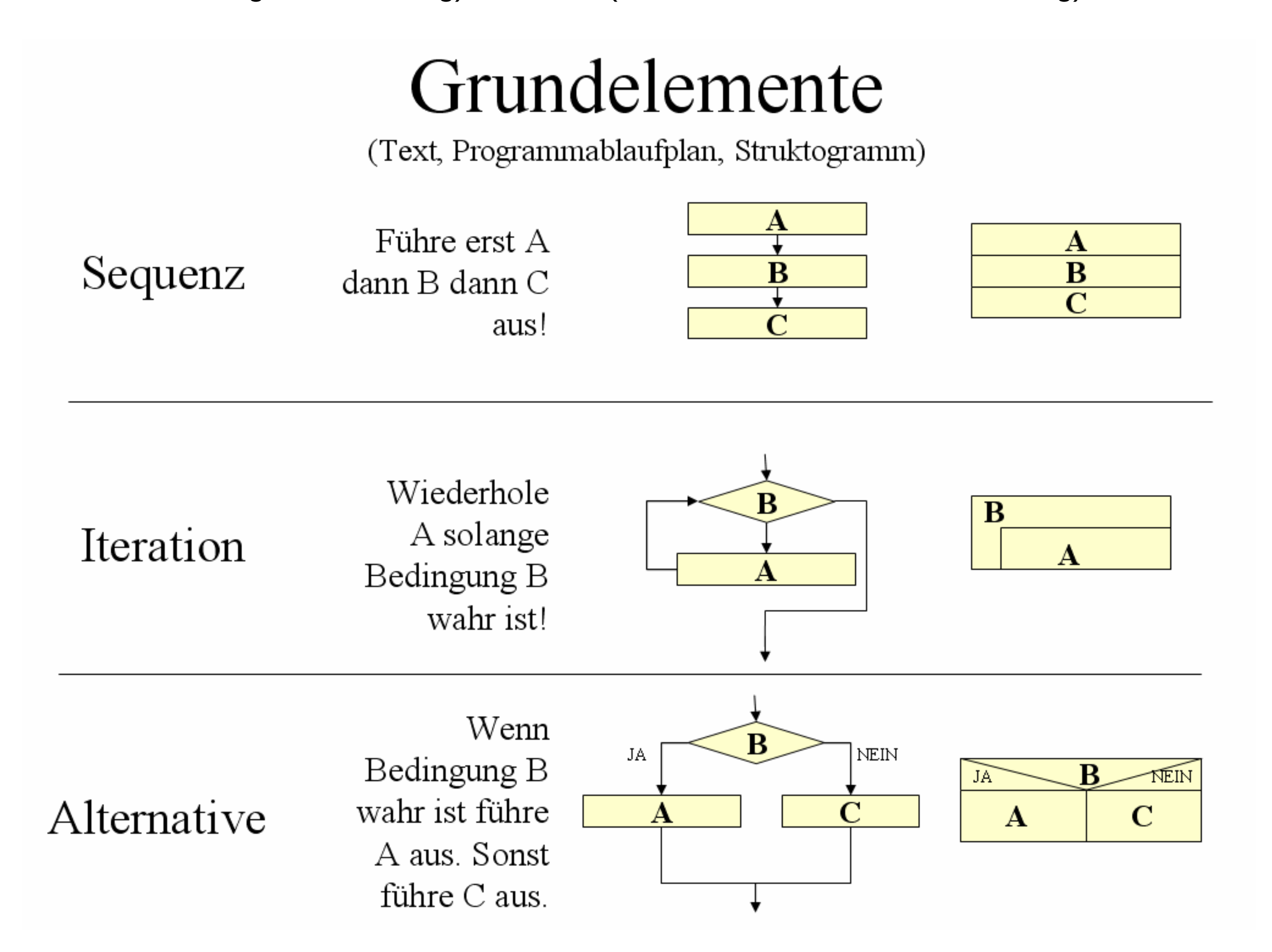
```
{
    ; Kennfelddaten in Adressregister Laden
    ldi r30,lo8 (Kennlinie1)
    ldi r31,hi8 (Kennlinie2)
    ...

}
```

11 Entwicklung eines Struktogramms mit SiSy AVR

11.1 Einleitung

Struktogramme (SG) oder auch Nassi-Shneiderman-Diagramme sind ein Entwurfsmittel der strukturierten Programmierung. Strukturiert meint in dem Fall, dass zur Realisierung eines Algorithmus auf das Verwenden von Sprunganweisungen (Goto, Jump) verzichtet wird. Für das Formulieren eines Algorithmus stehen dem Entwickler drei normierte Grundstrukturen zur Verfügung: Sequenz (Folge von Anweisungen), Alternative (Auswahl bzw. bedingte Anweisung), Iteration (Schleife, wiederholte Anweisung).

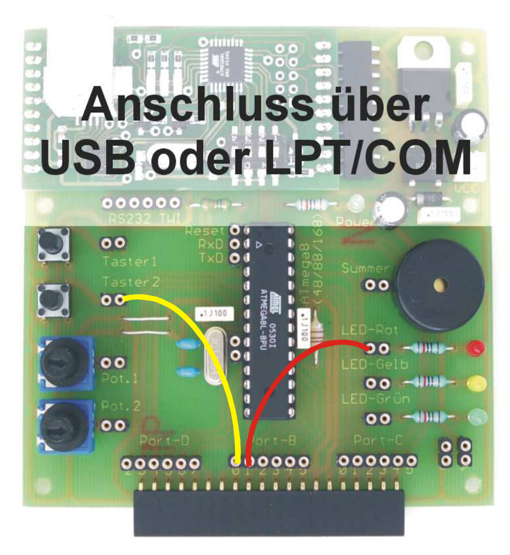


Struktogramme werden als Darstellungsmittel für strukturierte Sprachen wie C oder PASCAL verwendet, da hier im Gegensatz zu Assembler in der Regel auf Sprunganweisungen verzichtet wird.

11.2 Aufgabenstellung

Um die Arbeitsweise und den Umgang mit dem Struktogrammeditor in SiSy zu erlernen, realisieren Sie bitte das folgende Beispiel.

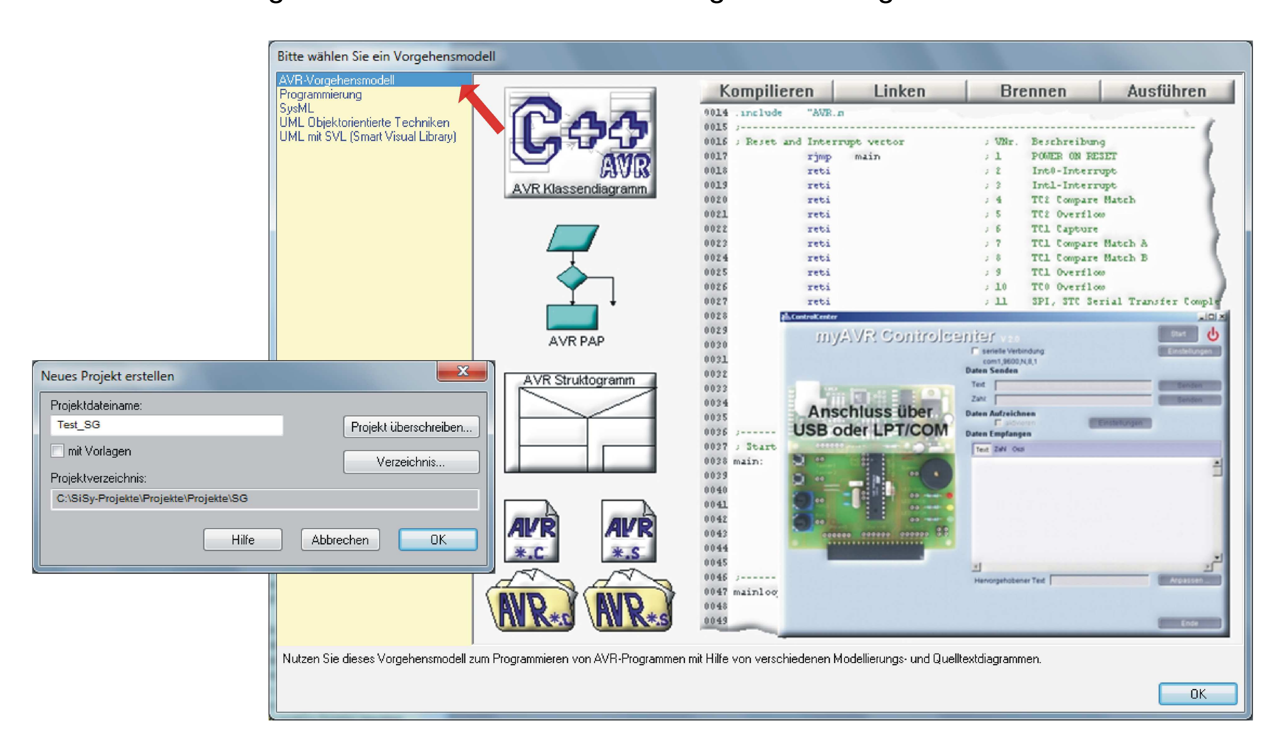
Es soll eine Mikrocontrollerlösung entwickelt werden, bei der auf Tastendruck eine LED eingeschaltet wird. Dabei soll der Taster an Port B 0 und die LED an Port B 1 angeschlossen werden



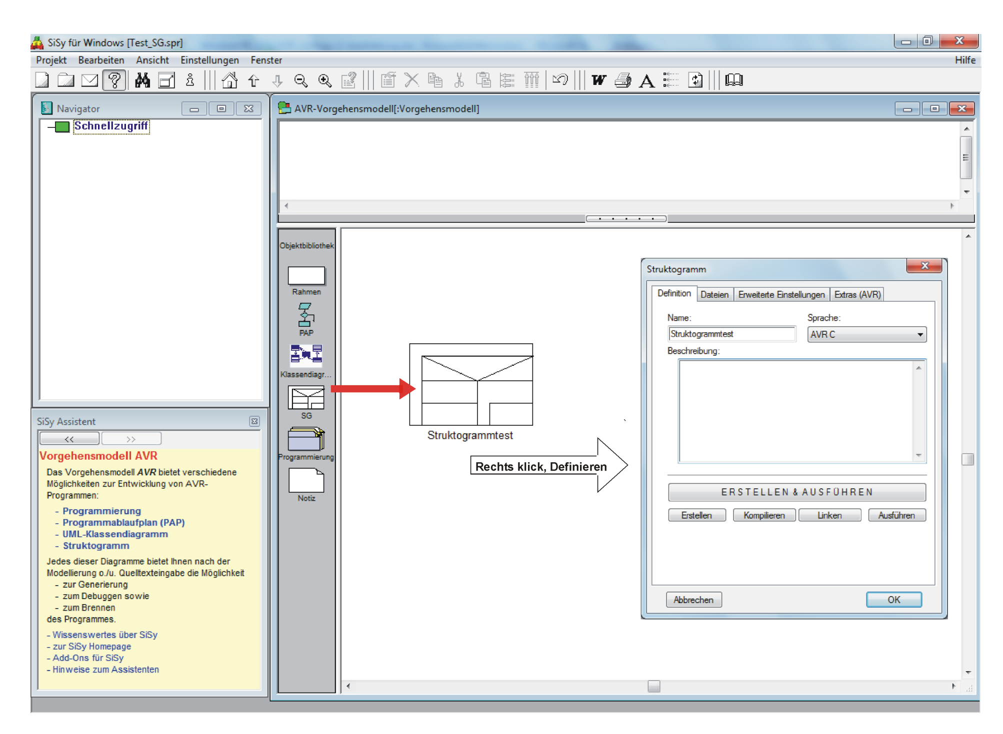
Beispiel für SG

11.3 Vorbereitung

Das Struktogramm kann je SiSy Projekt ein Hauptprogramm verwalten. Es ist also nötig, für jedes neue Struktogramm-Projekt auch ein neues SiSy-Projekt zu erzeugen. Starten Sie ggf. SiSy.
Erstellen Sie ein neues SiSy-Projekt (Hauptmenu -> Projekt -> Neu) mit dem Namen „Test_SG". Wählen Sie das Vorgehensmodell „AVR-Vorgehensmodell". Danach erscheint ein Dialogfenster mit verschiedenen Diagrammvorlagen.

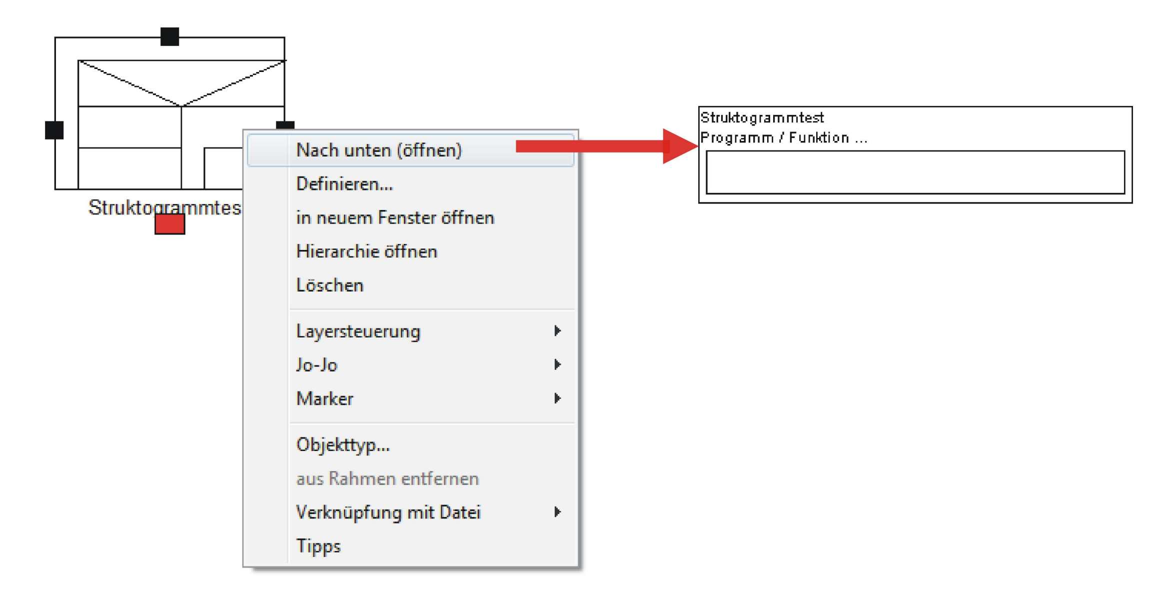


Legen Sie im weiteren Verlauf die AVR - Grundeinstellungen für das Projekt fest. Ziehen Sie ein Objekt vom Typ „SG" (Struktogramm) per Drag & Drop in das Diagrammfenster. Legen Sie den Namen für das Struktogramm fest.



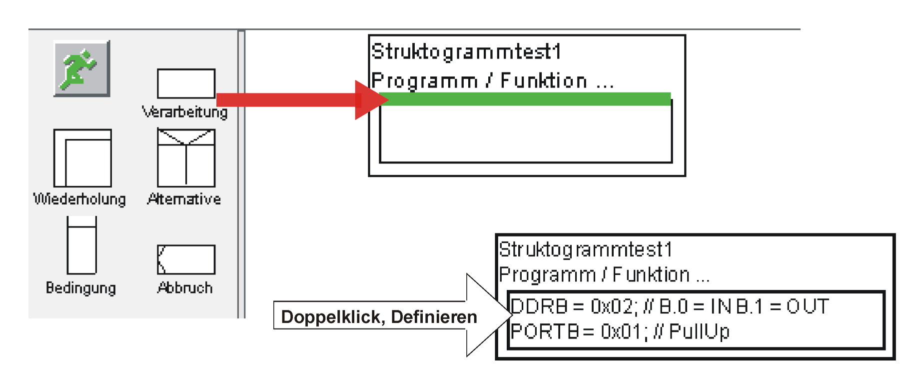
11.4 Struktogramm entwickeln

Um das Struktogramm zu entwickeln, muss das Struktogrammfenster geöffnet werden. Wählen Sie dazu auf dem Objekt rechte Maustaste -> Kontextmenü -> nach unten (öffnen).

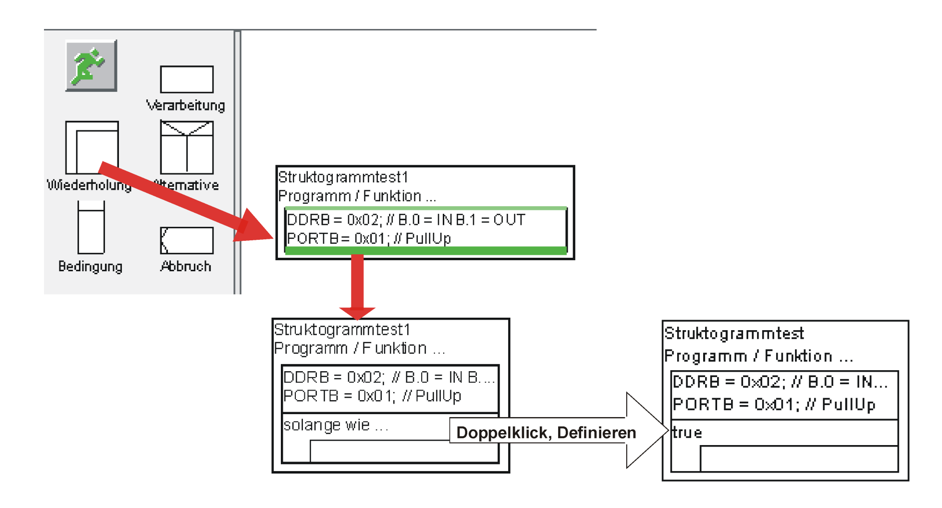


Führen Sie folgende Arbeitsschritte aus, um den Algorithmus für die oben genannte Aufgabe (10.2) zu entwerfen:

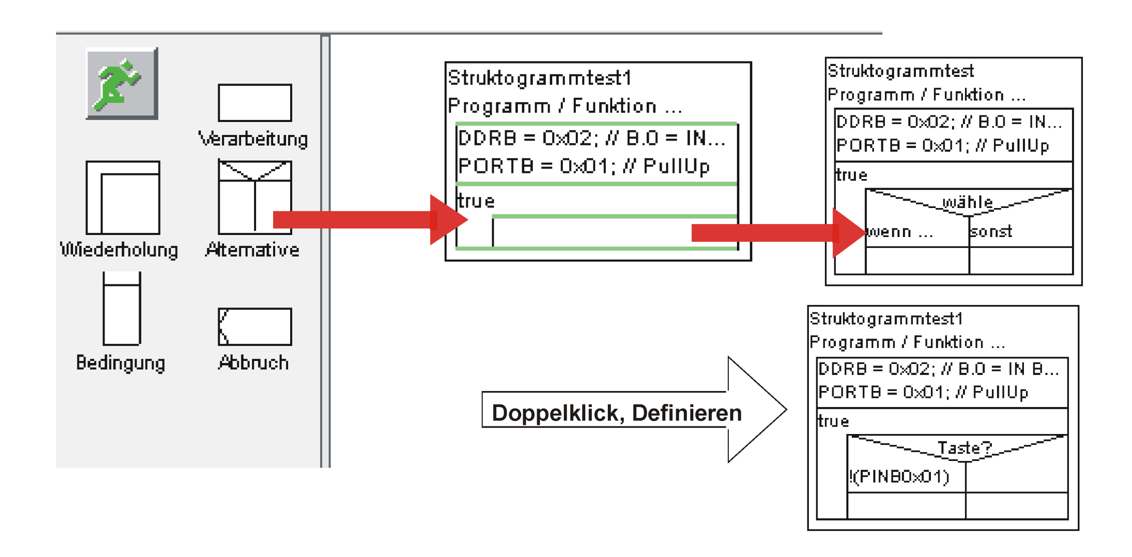
1. Konfigurieren Sie Port B 0 als Eingang und Port B 1 als Ausgang. Ziehen Sie dafür eine „Verarbeitungsroutine“ aus der Objektbibliothek in das Struktogramm.



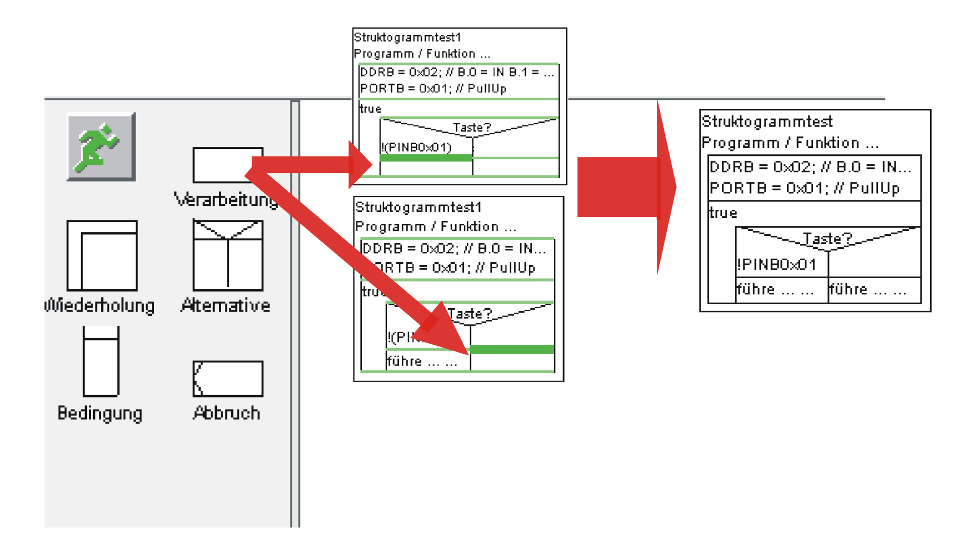
2. Fügen Sie eine „WHILE-Schleife“ ein.



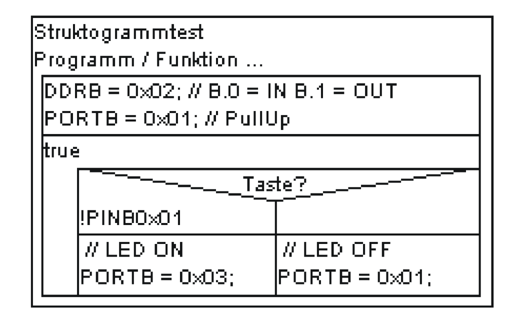
3. Ergänzen Sie die Mainloop durch eine „Alternative" per Drag and Drop. Achten Sie darauf, dass die Alternative innerhalb der Mainloop liegt. Geben Sie den Titel und die Bedingung für die Alternative ein.



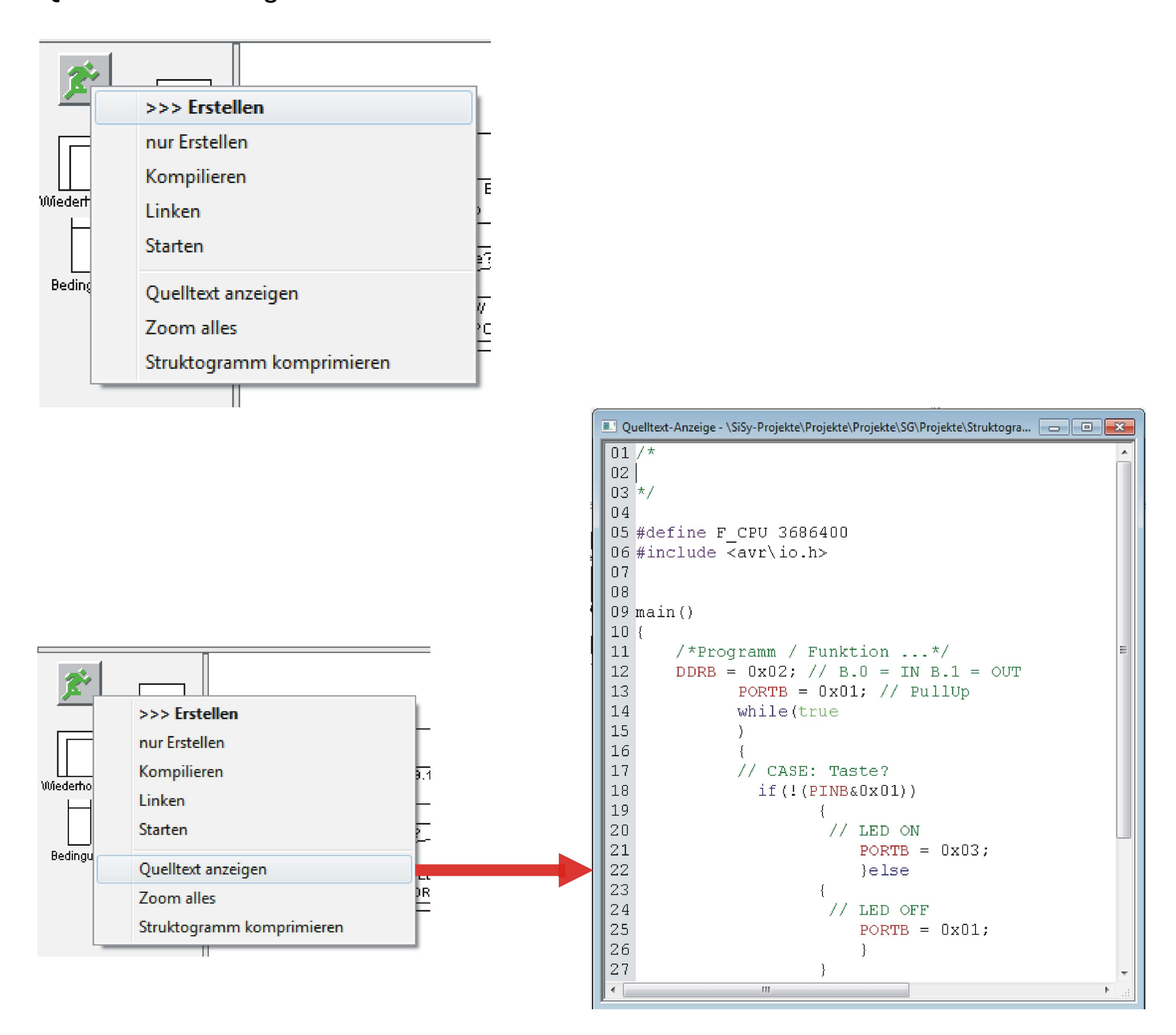
4. Als nächstes sind die Aktionen (DO-Elemente) in der Alternative zu ergänzen. Ziehen Sie dafür erneut jeweils eine „Verarbeitungsroutine" in die Alternative des Struktogramms.



5. Tragen Sie den C-Quellcode zum Ein- und Ausschalten der LED in die DO-Elemente ein. Selektieren Sie dazu das betreffende DO-Element per Doppelklick. Mit ESC verlassen Sie den Editiermodus.



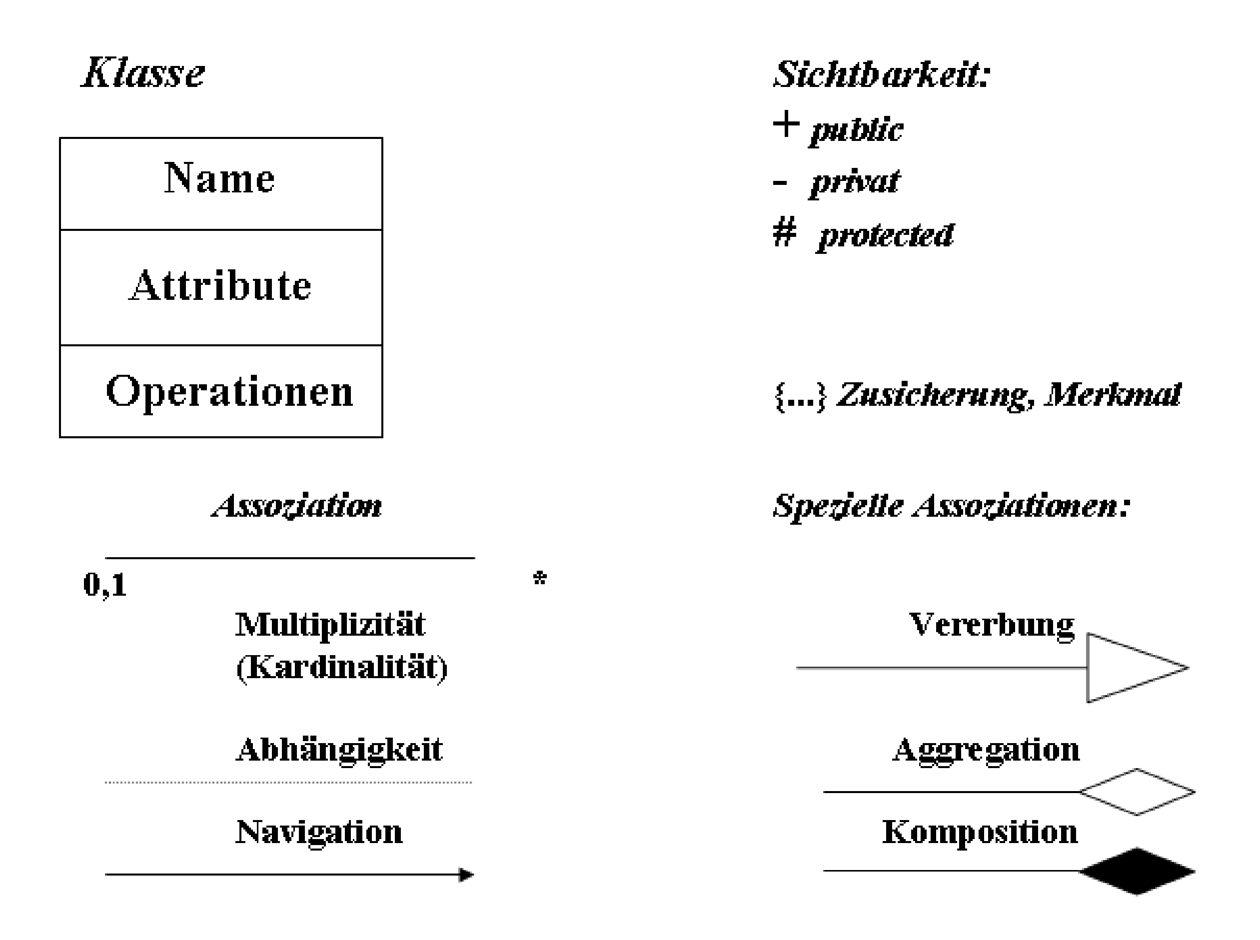
6. Generieren Sie den Quellcode für dieses Struktogramm und lassen Sie sich den Quellcode anzeigen.



12 Entwicklung eines Klassendiagramms mit SiSy AVR

12.1 Einleitung

Mit objektorientierten Programmiersprachen hat der Entwickler mächtige Sprachmittel, um komplexe Systeme realisieren zu können. C++ ist eine weit verbreitete objektorientierte Standardsprache. Als Visualisierungsmittel objektorientierter Programme gilt die international standardisierte Beschreibungssprache UML (Unified Modeling Language). SiSy AVR bietet dem Entwickler das UML Klassendiagramm mit Codegenerierung für AVR C++. Der folgende Abschnitt beschreibt die Handhabung des Klassendiagramms in SiSy AVR. Die Abbildung zeigt Ihnen eine Kurzübersicht der Modellierungselemente des UML Klassendiagramms.



Kurzübersicht Elemente des UML Klassendiagramms

Schreibweise von Attributen:
Attribute beginnen mit einem Kleinbuchstaben.

```
Sichtbarkeit name : Typ = Initialwert {Merkmal}
# temperatur : uint8_t = 25
```

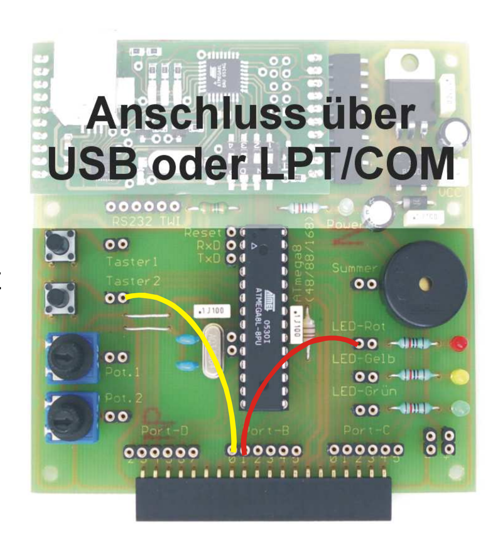
Schreibweise von Operationen:
Operationen beginnen mit einem Kleinbuchstaben.

```
Sichtbarkeit name (Parameter:Typ = Standardwert, ...) : Rückgabetyp {Merkmal}
+ setTemperatur ( temp : integer = 25 ) : bool
```

12.2 Aufgabenstellung

Um die Arbeitsweise und den Umgang mit dem Klassendiagramm in SiSy zu erlernen, realisieren Sie bitte das folgende einfache Beispiel.

Es soll eine Mikrocontrollerlösung entwickelt werden, bei der auf Tastendruck eine LED eingeschaltet wird. Dabei soll der Taster an Port B 0 und die LED an Port B 1 angeschlossen werden

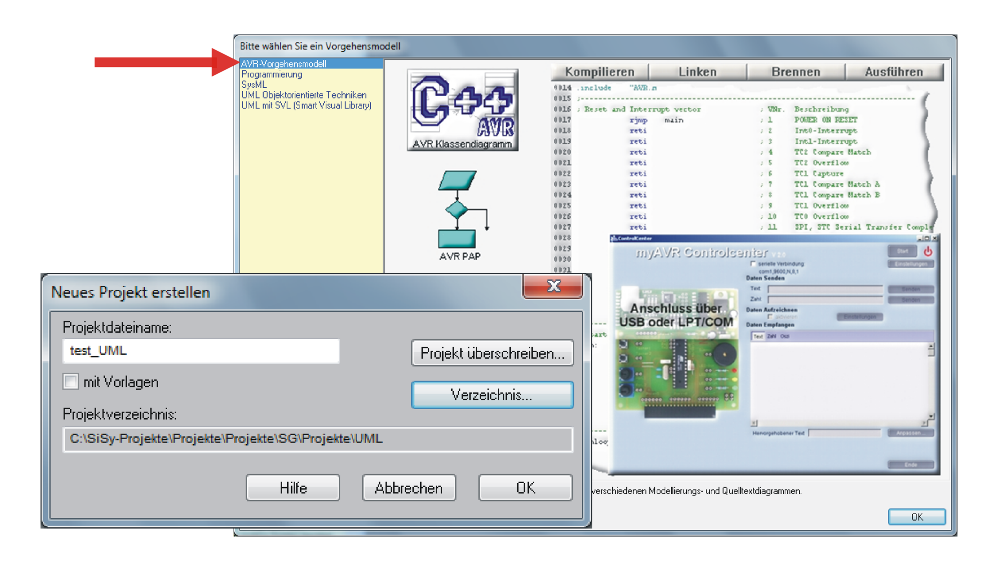


UML Beispiel

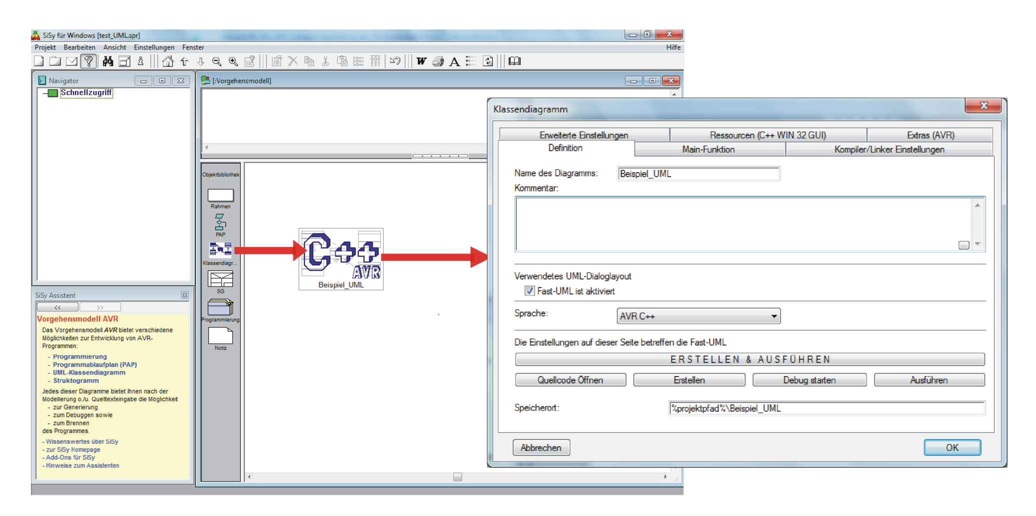
12.3 Vorbereitung

Starten Sie ggf. SiSy.

Erstellen Sie ein neues SiSy-Projekt (Hauptmenu -> Projekt -> Neu) mit dem Namen „Test_UML". Wählen Sie das Vorgehensmodell „AVR-Vorgehensmodell". Danach erscheint ein Dialogfenster mit möglichen Diagrammvorlagen.

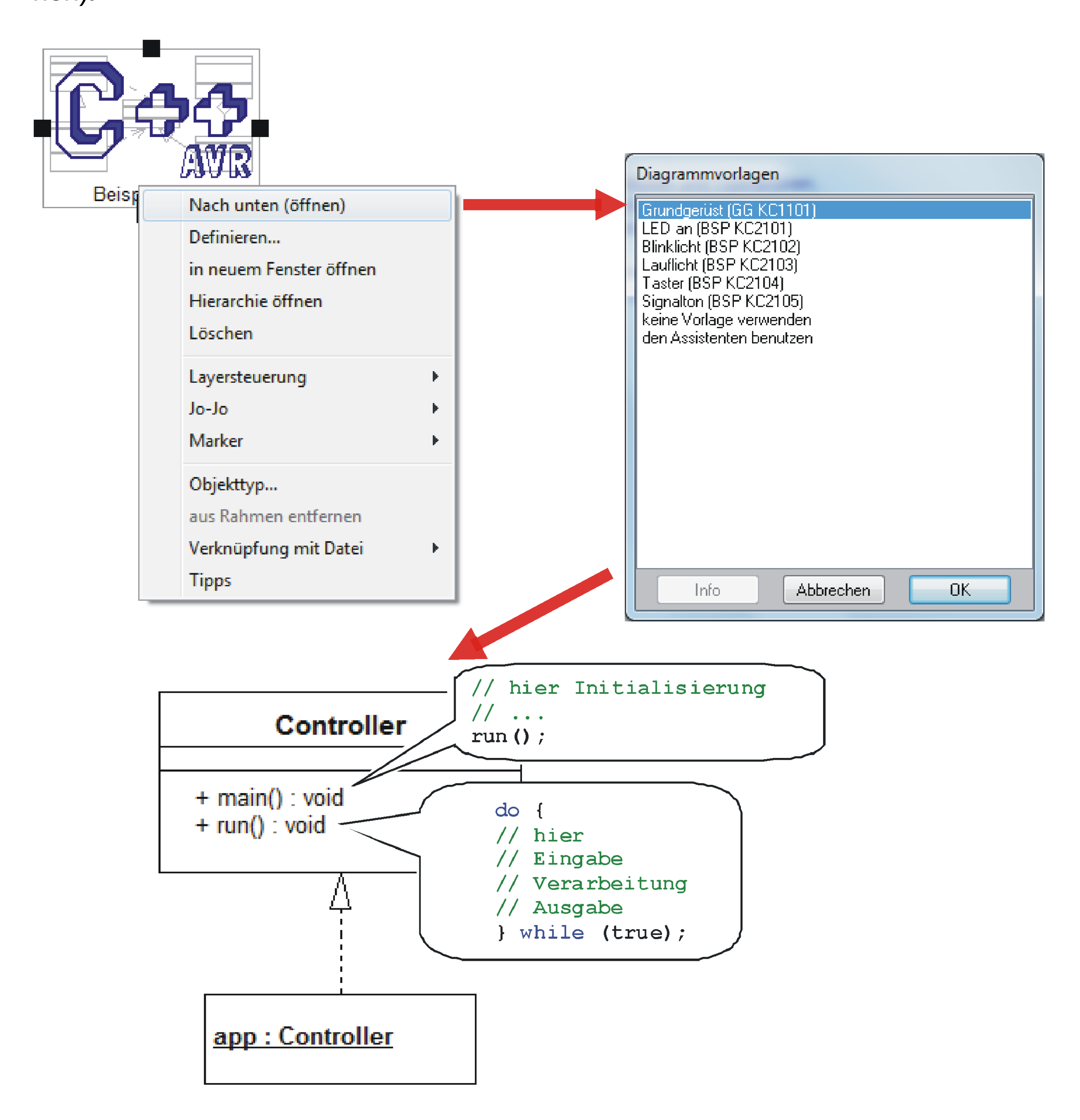


Legen Sie im weiteren Verlauf die AVR-Grundeinstellungen für das Projekt fest. Ziehen Sie ein Objekt vom Typ „Klassendiagramm" per Drag & Drop in das Diagrammfenster. Legen Sie den Namen für das Klassendiagramm fest.



12.4 Grundstruktur laden

Um das Klassenmodell zu entwickeln, muss das Klassendiagramm geöffnet werden. Wählen Sie dazu auf dem Objekt rechte Maustaste -> Kontextmenü -> nach unten (öffnen).



Die Struktur einer objektorientierten Mikrocontrollerlösung in SiSy erfordert im Klassenmodell eine Applikationsklasse (Hauptklasse), die sich dadurch auszeichnet, dass diese über eine Methode (Operation) mit dem Namen **main** verfügt. Der Codegenerator erzeugt eine Instanz diese Klasse und ruft die Main-Methode auf.

```
#define cpp_Test_UML
#define F_CPU 3686400
#include <avr\io.h>

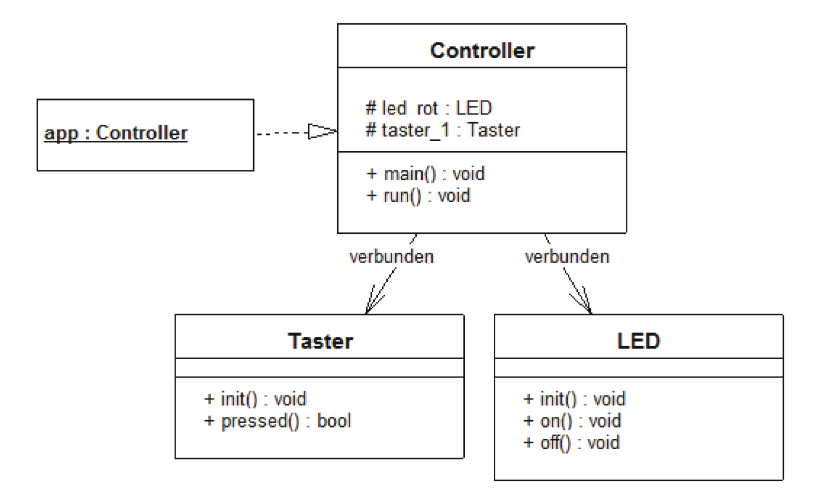
main (void)
{
    Controller MyApp;
    MyApp.main();
}
```

12.5 Systemstruktur entwerfen

Die Systemstruktur einer objektorientierten Anwendung bildet die Objekte und deren Beziehungen im Programm ab, welche im realen System als physische Elemente vorhanden sind. Als Bauplan der Objekte dienen Klassendeklarationen, welche die Eigenschaften (Attribute) und das Verhalten (Methoden/Operationen) der Objekte beschreiben. Das Klassendiagramm beschreibt also die Struktur der Klassen (Baupläne der Objekte) und die Beziehungen zwischen den Klassen. In unserer Aufgabenstellung finden wir die Objekte des Systems als **Substantive**, deren Beziehungen und Verhalten als **Verbalphrasen**.

*Es soll eine Mikro**controller**lösung entwickelt werden, bei der durch **drücken** eines **Tasters** eine **LED** **eingeschaltet** wird. Dabei soll der Taster an Port B 0 und die LED an Port B 1 **angeschlossen** werden.*

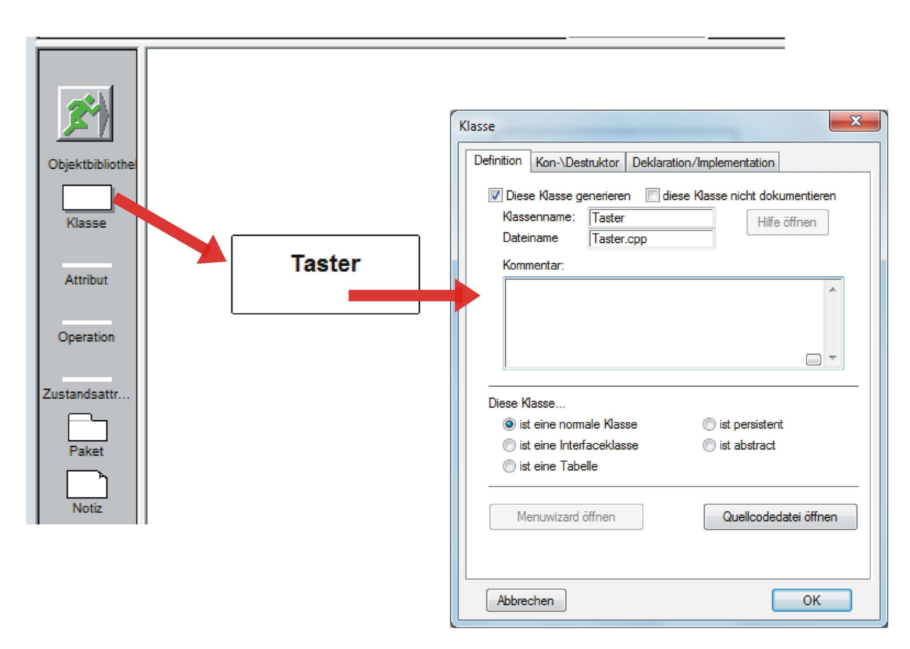
Daraus lässt sich folgende Klassenstruktur ableiten:



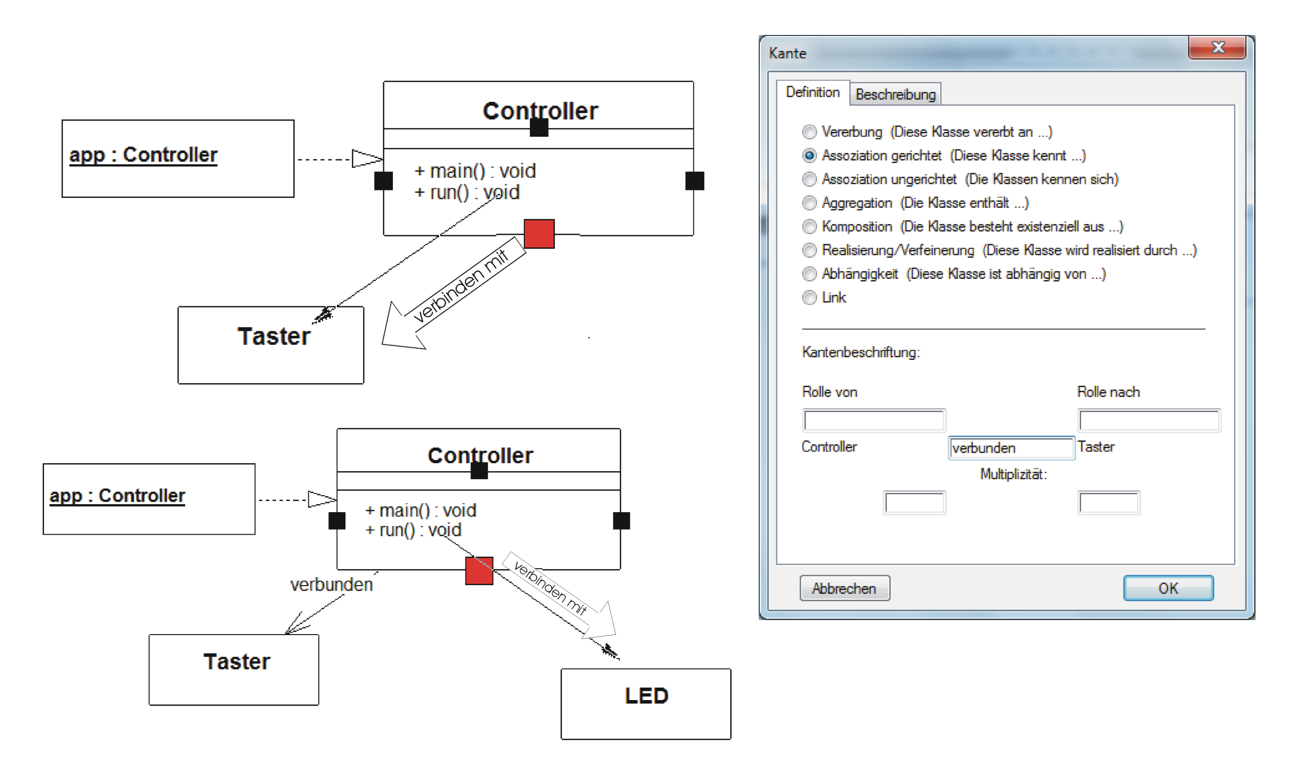
Klassenstruktur, Systementwurf mit dem UML Klassendiagramm

Zum Erstellen dieses Klassenmodells sind folgende Arbeitsschritte nötig:

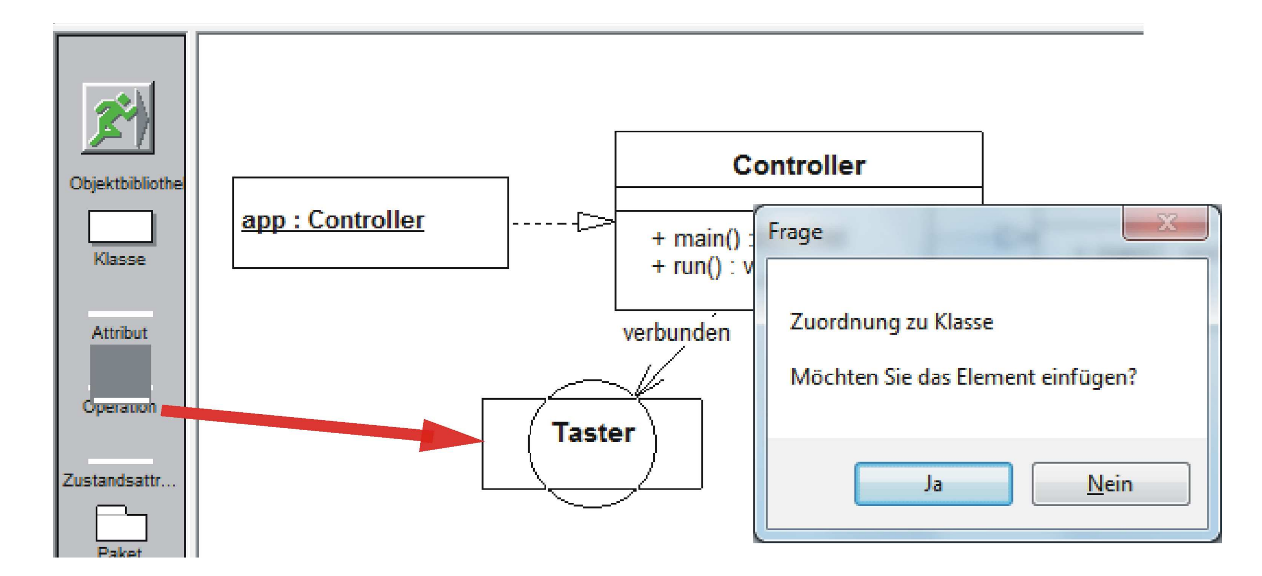
1. **Klassen einfügen und definieren**, ziehen Sie dazu das Objekt vom Typ „Klasse“ per Drag & Drop aus der Objektbibliothek in das Diagramm. Definieren Sie den Namen der Klasse und setzen die Option „diese Klasse generieren“.



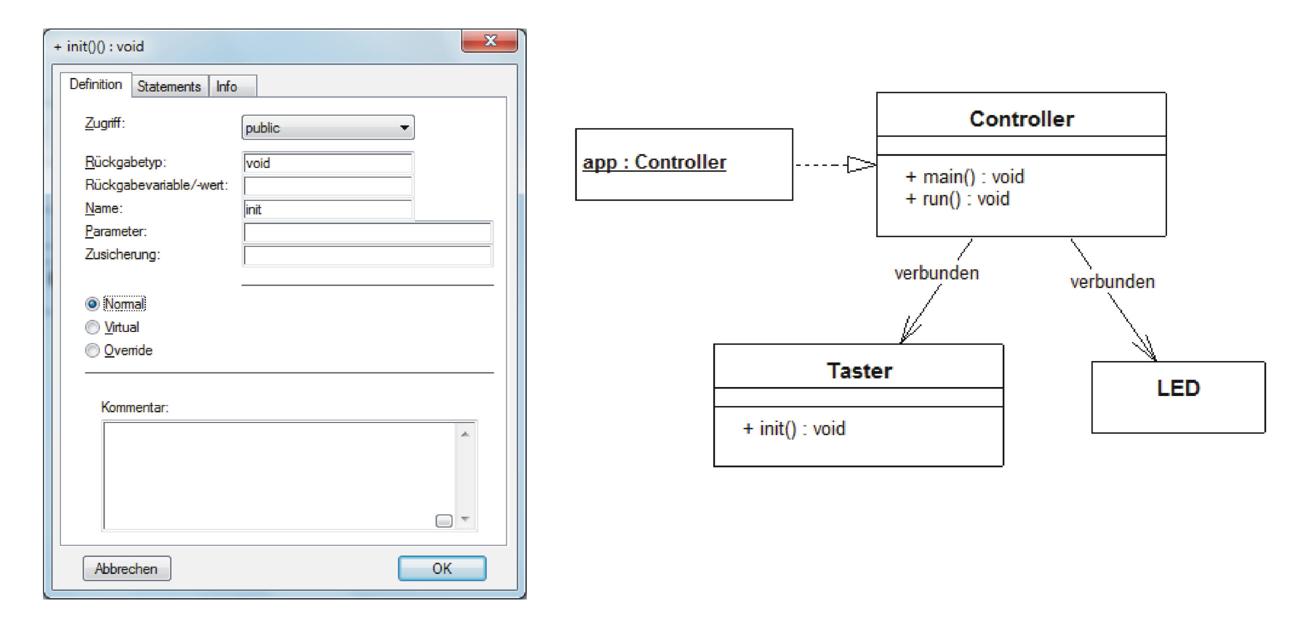
2. **Klassen verbinden**, selektieren Sie die Klasse, von der aus eine Verbindung gezogen werden soll. Ziehen Sie per Drag & Drop ausgehend vom roten Verteiler eine Verbindung auf die gewünschte Klasse. Wählen Sie den Verbindungstyp, zum Beispiel „Assoziation gerichtet“ und beschriften Sie die Verbindung.



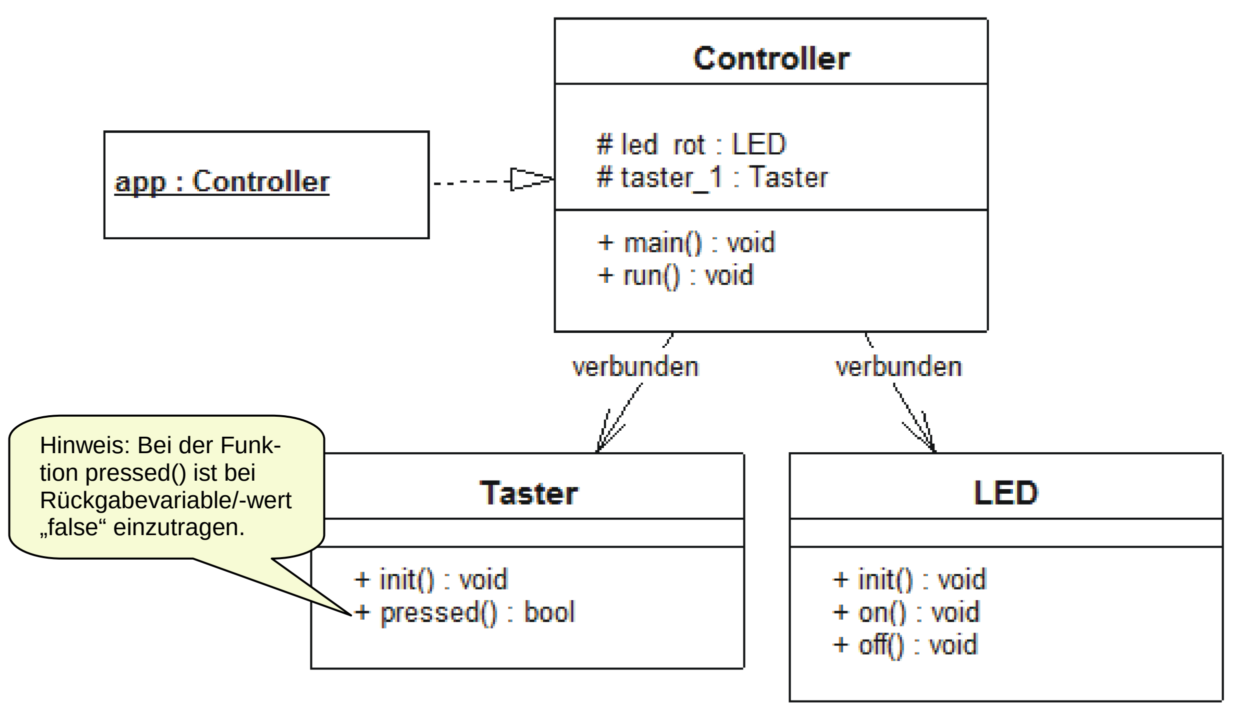
3. **Operationen und Attribute einfügen und definieren**, ziehen Sie dazu ein Objekt vom Typ „Operation“ oder „Attribut“ aus der Objektbibliothek auf die Klasse, in die das Attribut oder die Operation eingefügt werden soll. Bestätigen Sie die Sicherheitsabfrage zum Einfügen des Elementes.



Definieren Sie Zugriff (Sichtbarkeit), Name, Typ und Parameter der Operation bzw. Zugriff (Sichtbarkeit), Name, Typ und Initialwert des Attributes.



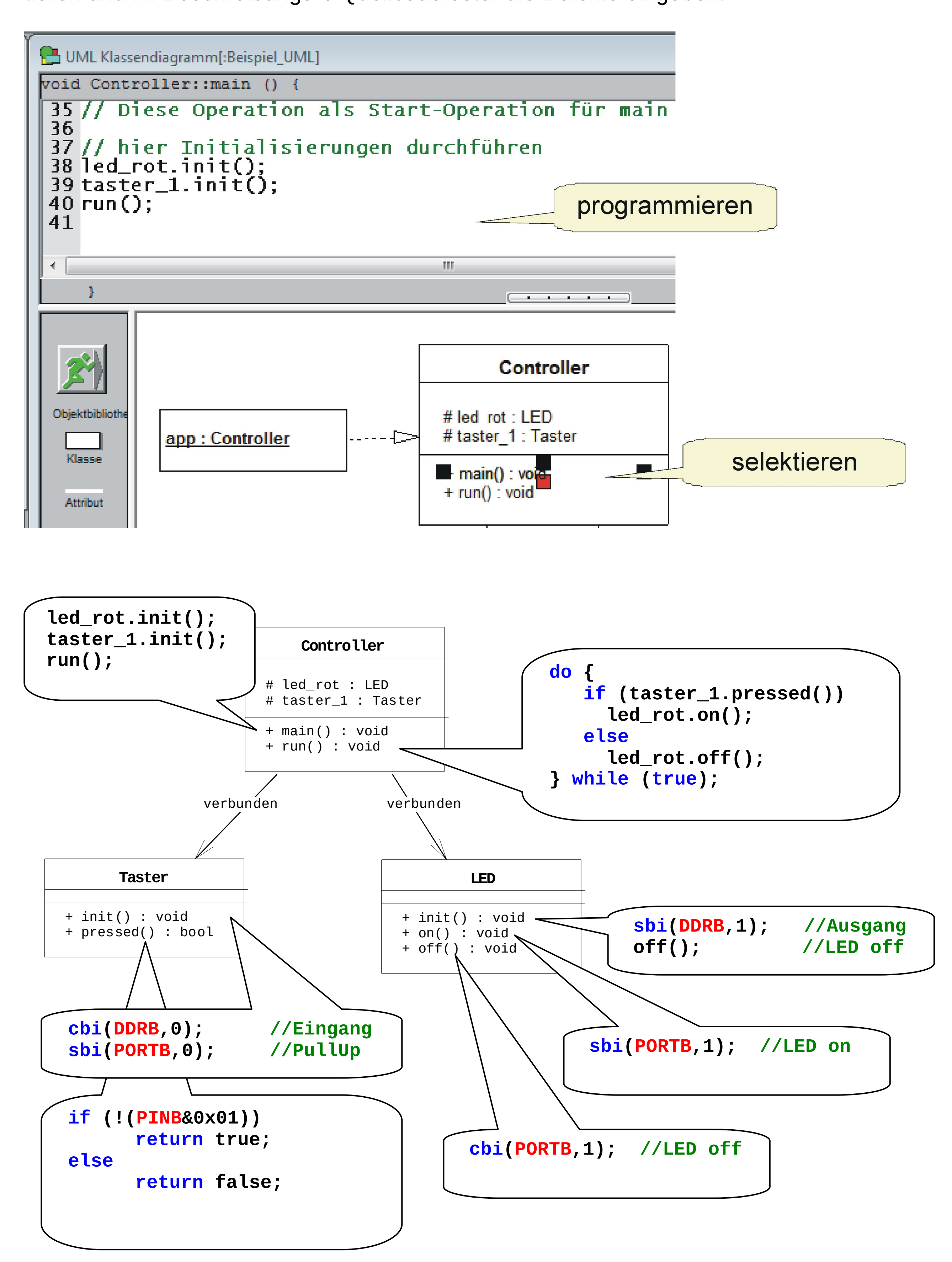
Vervollständigen Sie das Klassenmodell entsprechend der Abbildung.



endgültiger Systementwurf mit dem UML Klassendiagramm

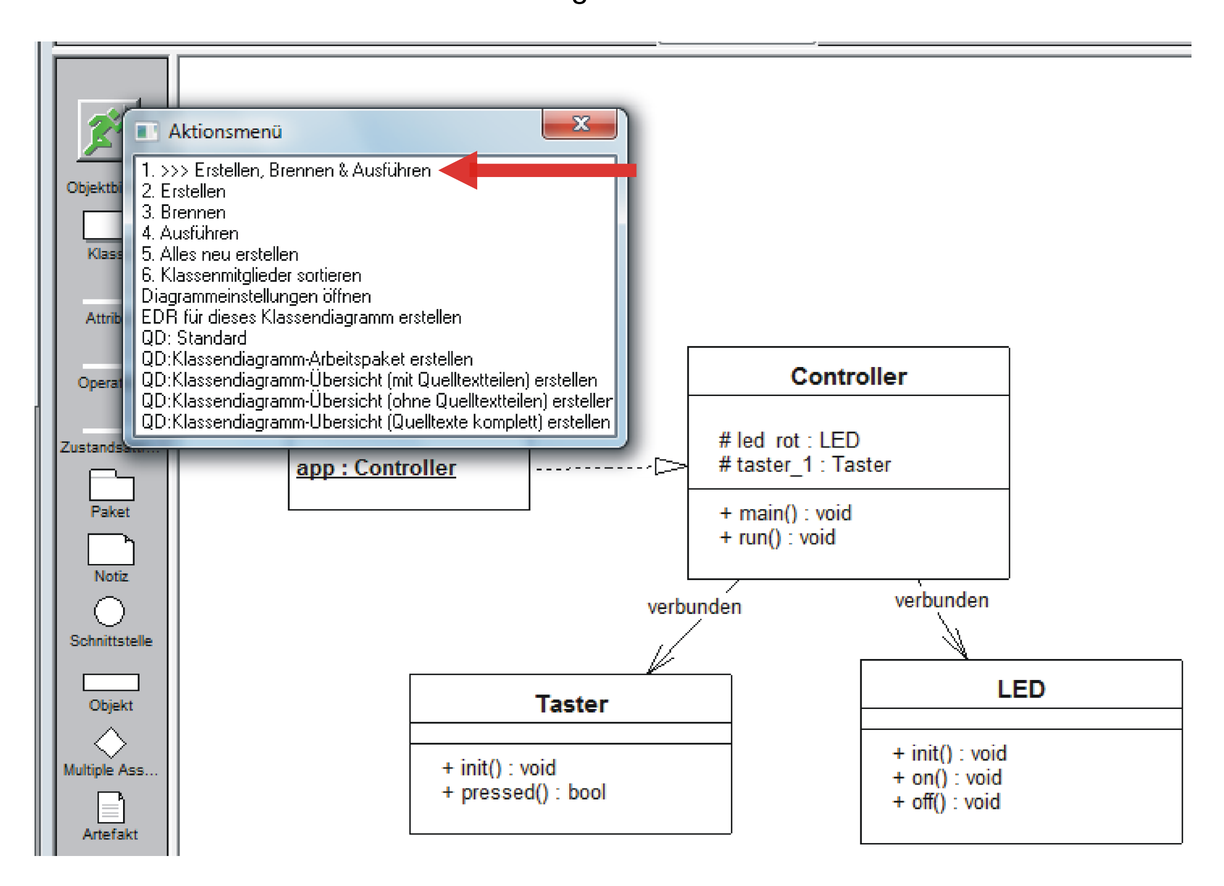
12.6 Systemverhalten programmieren

Die Operationen müssen mit der entsprechenden Logik versehen werden. Sie können den Quellcode der Operationen bearbeiten, indem Sie die gewünschte Operation selektieren und im Beschreibungs- / Quellcodefester die Befehle eingeben.

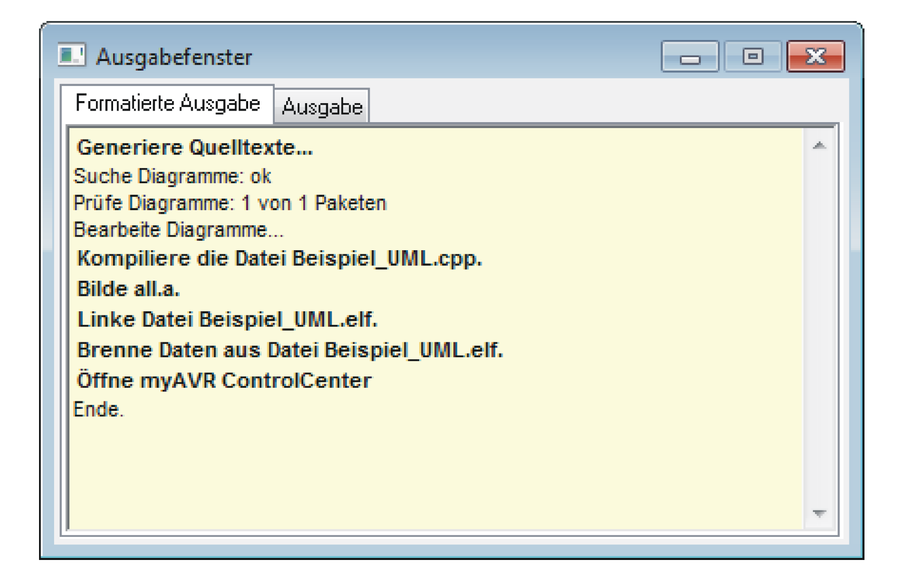


12.7 Übersetzen, Brennen und Testen

Die Codegenerierung aus dem UML Klassendiagramm, das Kompilieren, Linken und Brennen kann über das Aktionsmenü gestartet werden.



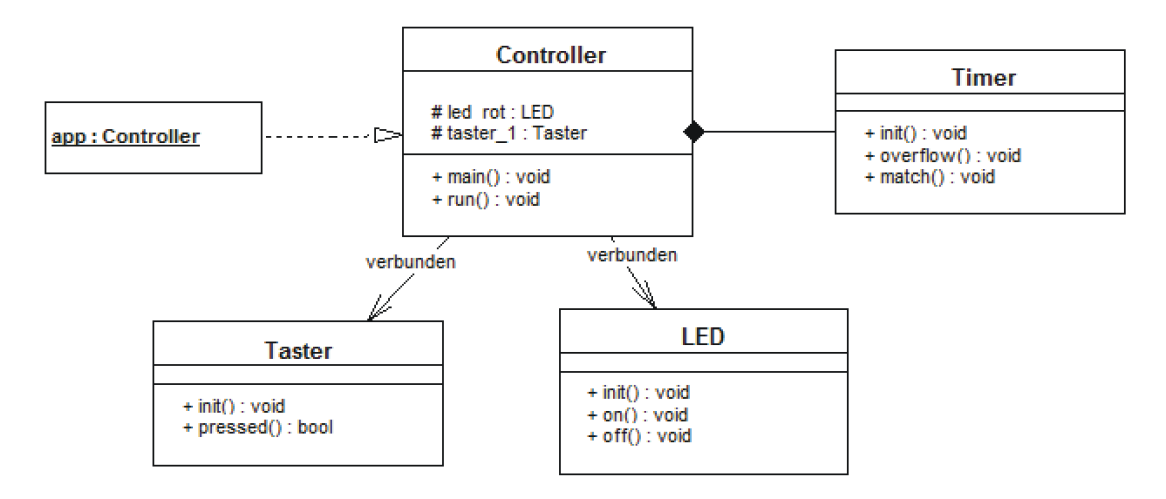
Sie erhalten im Ausgabefenster ein Protokoll der ausgeführten Aktionen.



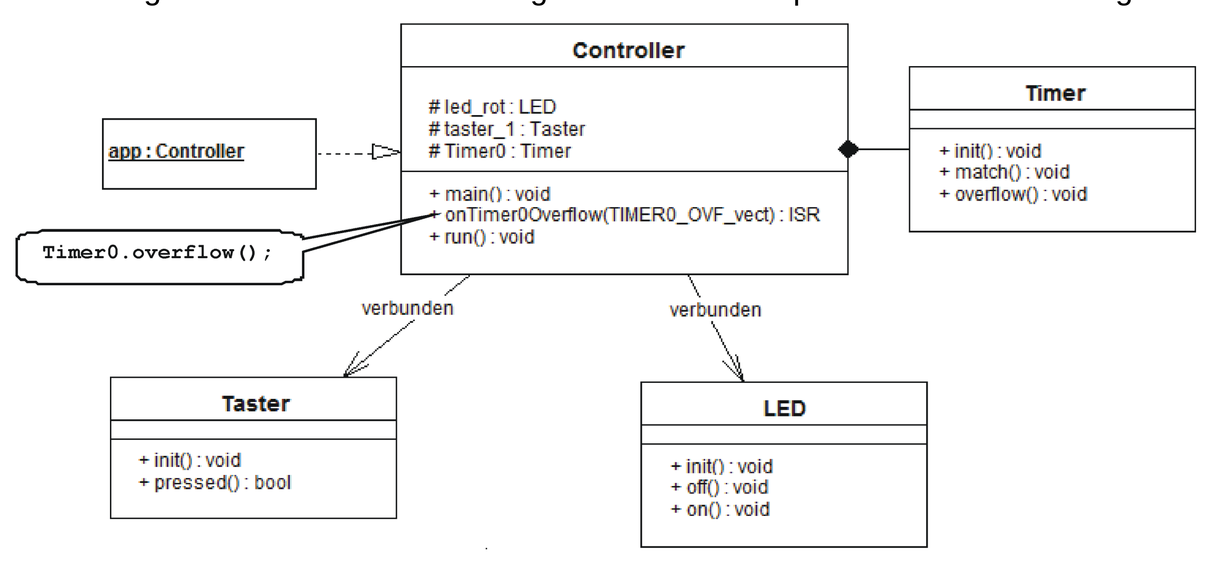
12.8 Interrupt-Service-Routinen (ISR) im Klassendiagramm

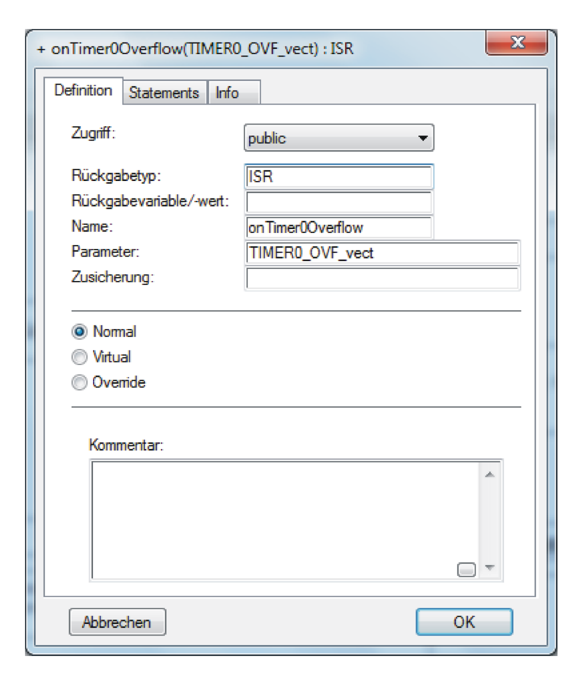
Interrupt-Service-Routinen (im weiteren ISR) werden in AVR C++ durch das Schlüsselwort ISR gekennzeichnet. Sie bilden eigenständige Funktionen. Die Besonderheit bei der Realisierung einer ISR liegt darin, dass es sich hier um ein C-Makro handelt und nicht um eine echte Funktion. Diese können also keine Methode einer Klasse sein. Um eine ISR im Klassendiagramm zu realisieren gehen Sie wie folgt vor (Erweiterung des Beispiels um die Klasser Timer und der `ISR Timer0 Overflow`):

1. Die Klasse für die interruptfähige Komponente modellieren und die Methode einfügen, welche beim Interrupt ausgeführt werden soll. In unserem Beispiel ist es die Klasse **`Timer`** mit der Methode **`overflow`**. Die Initialisierung des Timers muss entsprechend erfolgen.



2. Ergänzen Sie das Klassendiagramm weiter entsprechend der Abbildung.





Beispiel:

```
#include  <io.h>

// Instanz der interruptfähigen Klasse anlegen
Timer Timer0;

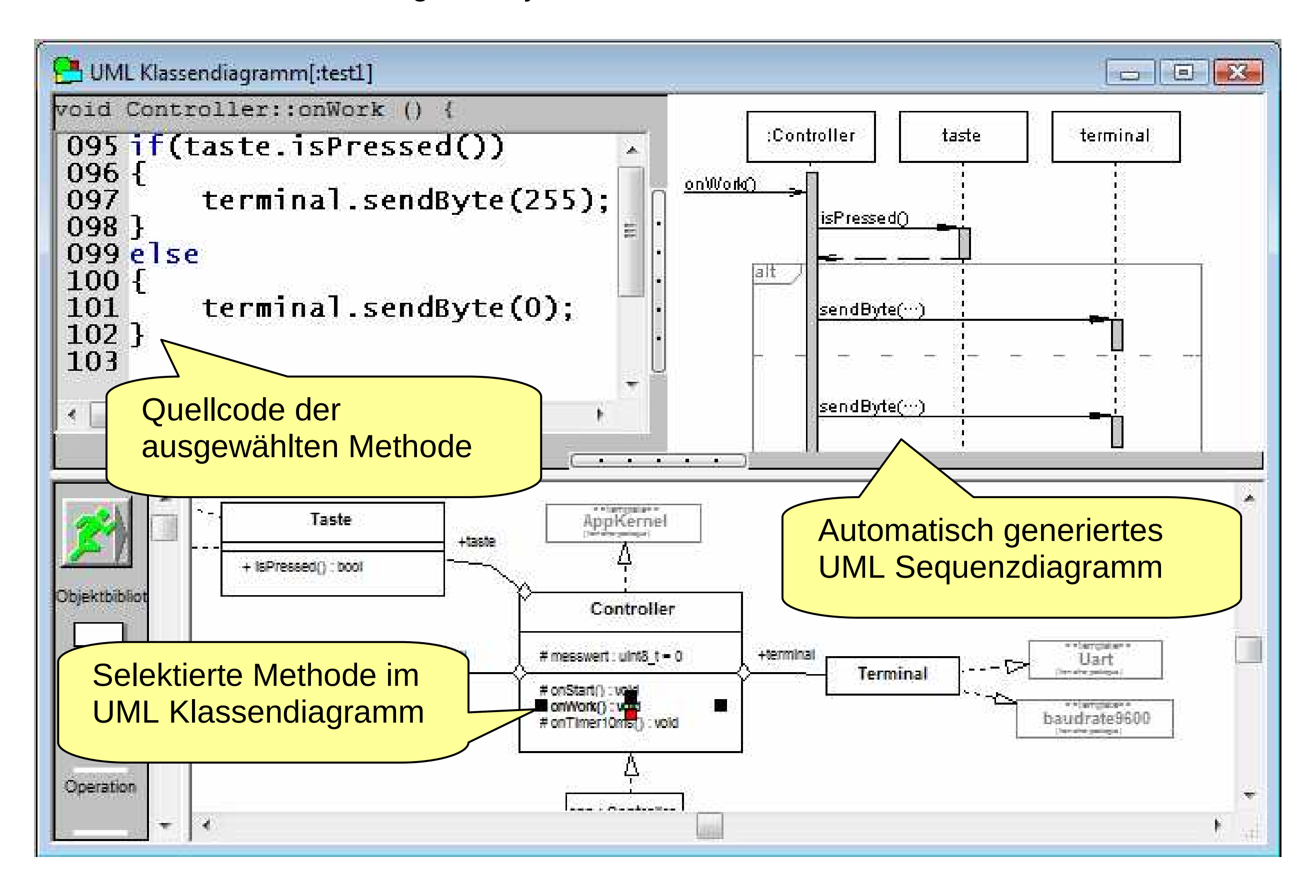
// Interuptmakro mit aufruf der betreffenden Methode
ISR (TIMER0_OVF_vect)
{
     Timer0.overflow();
}
```

In der Interrupt-Service-Routine rufen Sie einfach die Methode der betreffenden Klasse auf.

13 Der Sequenzdiagrammgenerator

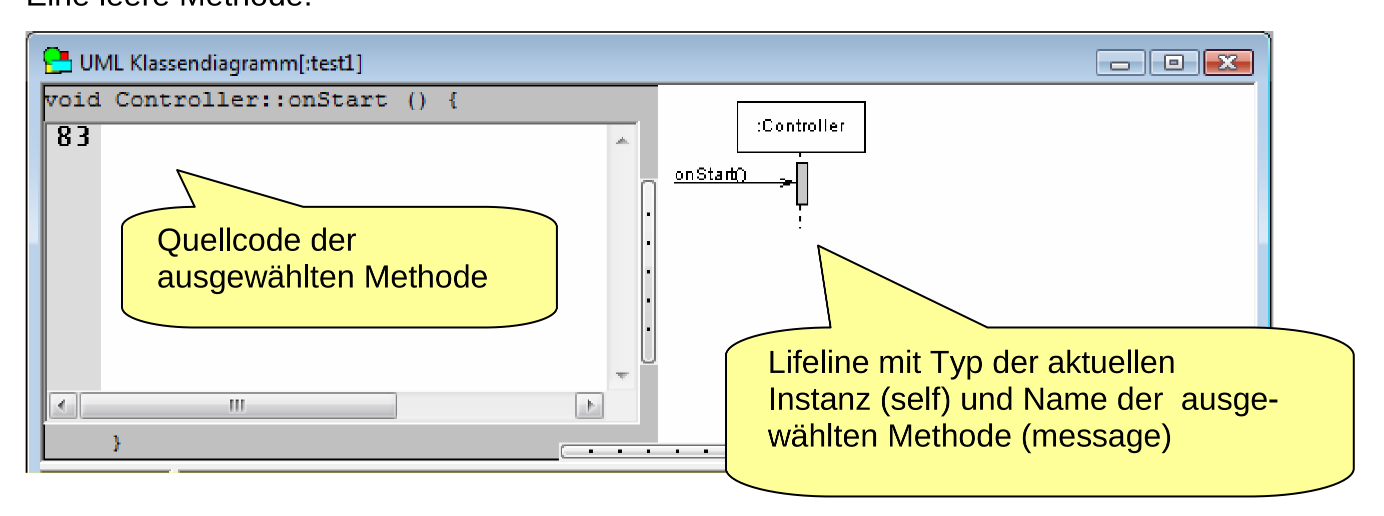
13.1 Einführung in das Sequenzdiagramm

Sequenzdiagramme sind eine wichtige Darstellungstechnik der UML für die Dokumentation des Systemverhaltens und der Interaktion von Objekten. SiSy verfügt über die Möglichkeit aus objektorientiertem Programmcode, wie zum Beispiel dem C++ Quellcode einer Klassenmethode, automatisch das entsprechende Sequenzdiagramm zu generieren. Dabei wird bereits während der Eingabe des Quellcodes simultan das Sequenzdiagramm erstellt. Der Entwickler ist damit in der Lage, die Reihenfolge der Nachrichten und die beteiligten Objekte mit deren Lebenszeit zu überblicken.

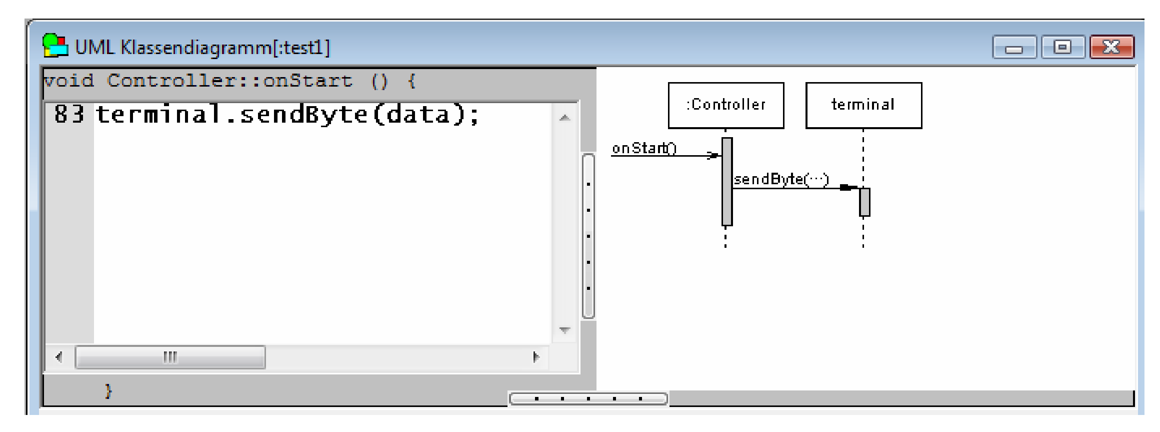


13.2 Sequenzen

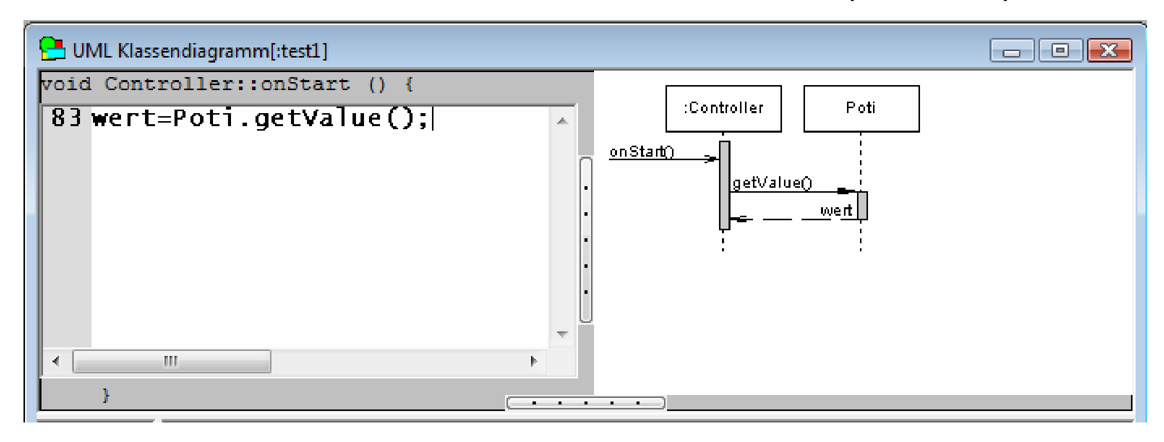
Eine leere Methode:



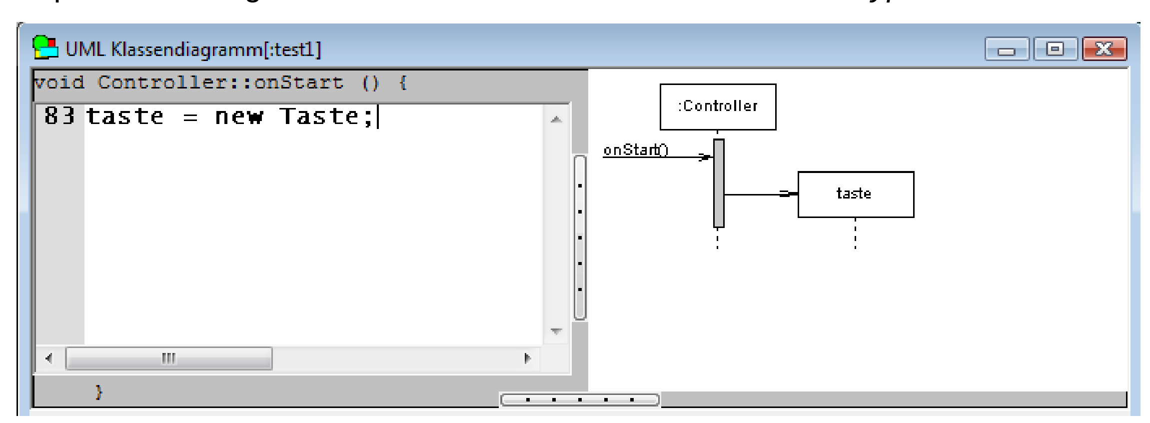
Eine einfache synchrone Nachricht: *Instanz.Nachricht (Parameter);*



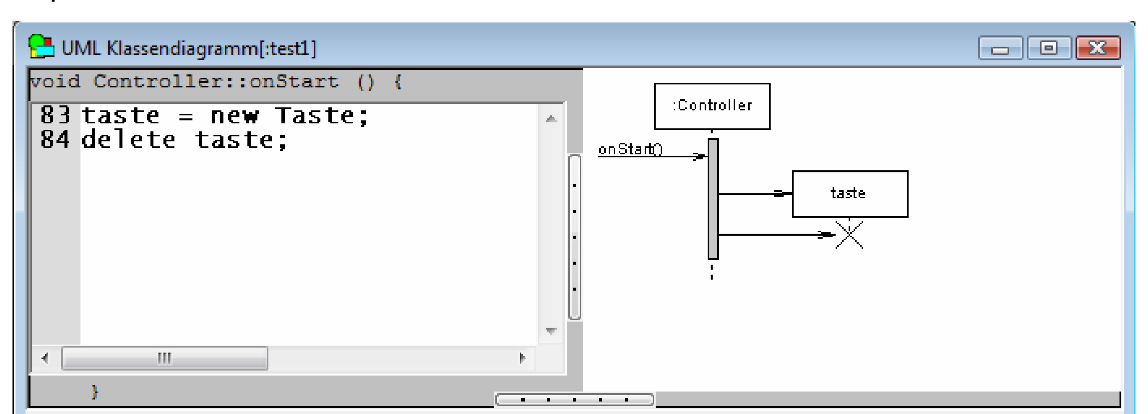
Eine Nachricht mit Antwort: *Variable = Instanz.Nachricht (Parameter);*



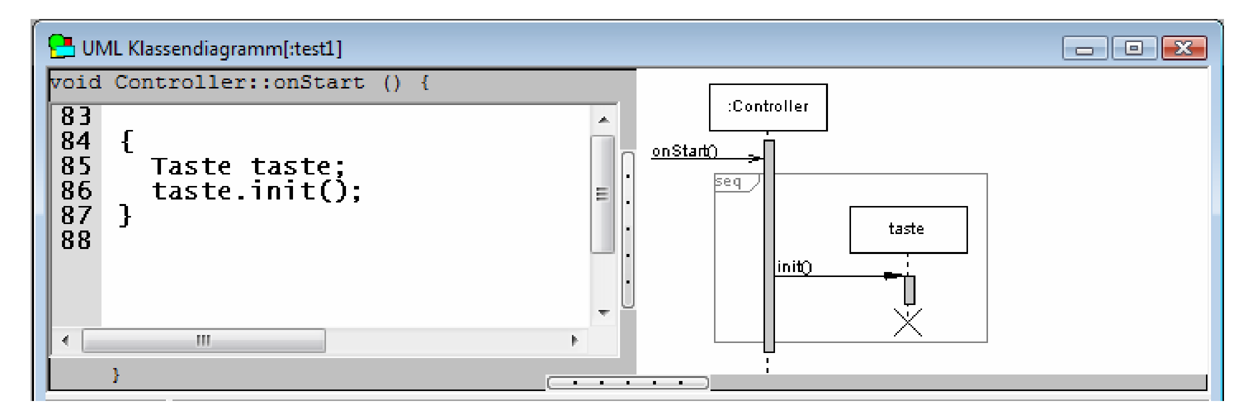
Explizites Erzeugen einer Instanz mit new: *Instanz = new Typ;*



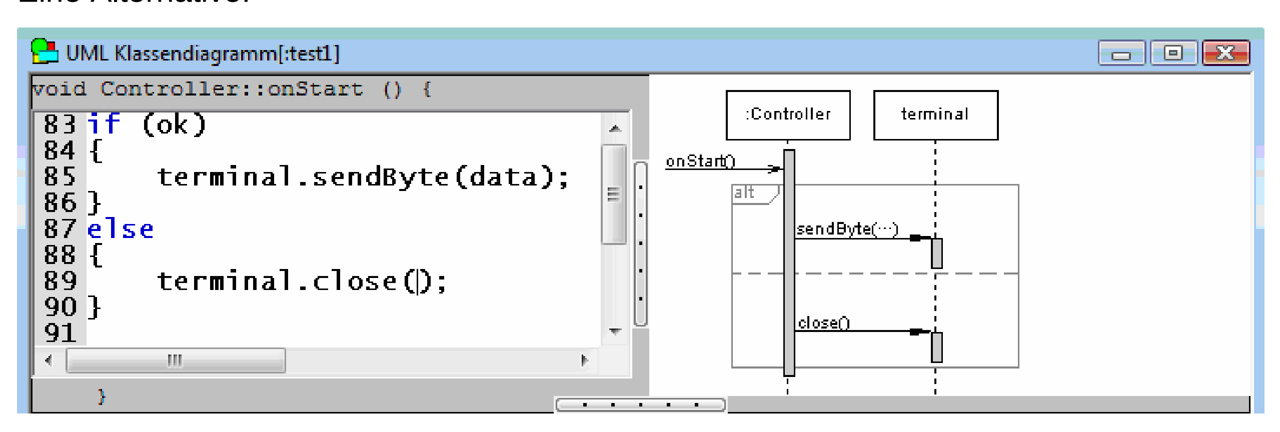
Explizites Zerstören einer Instanz mit delete: *delete Instanz;*



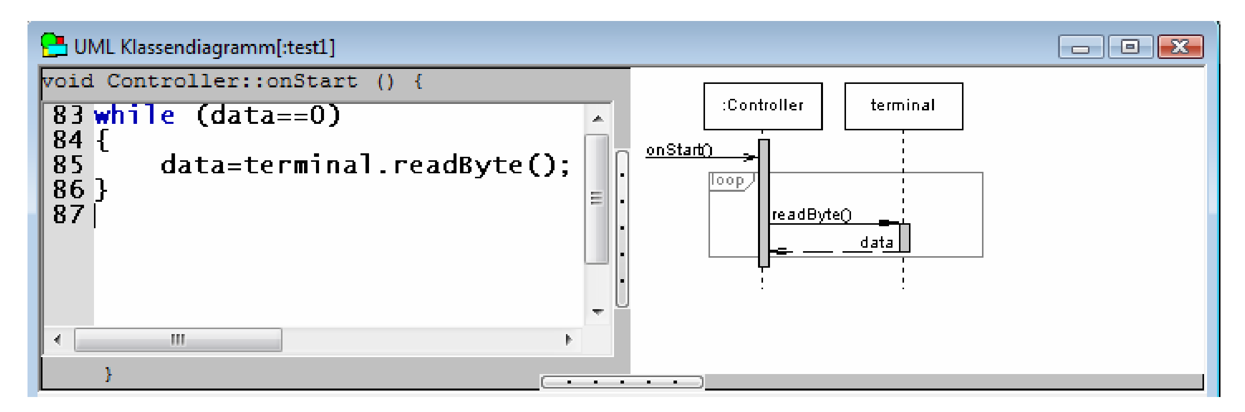
Eine lokale Instanz:



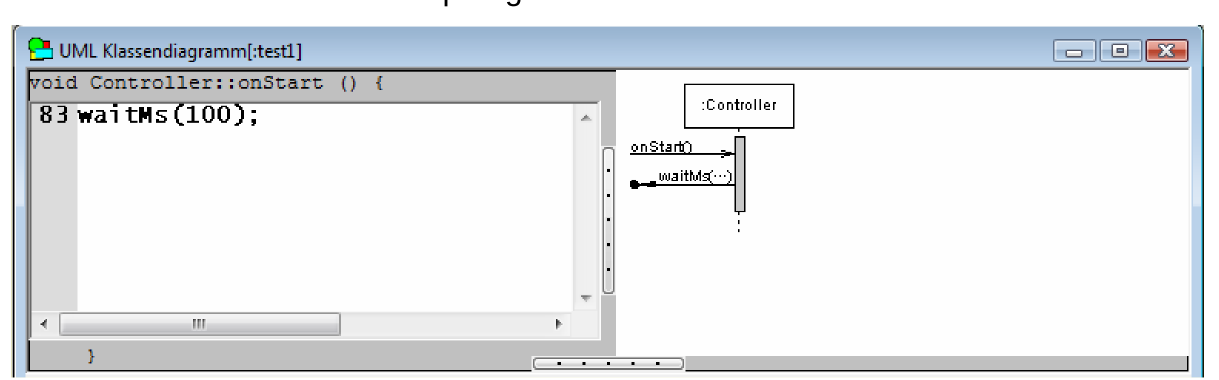
Eine Alternative:



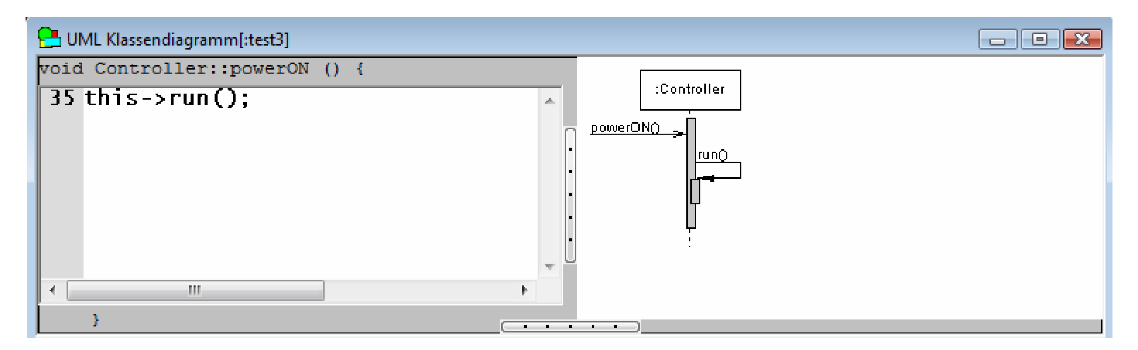
Eine Schleife:



Nachricht an unbekannten Empfänger:

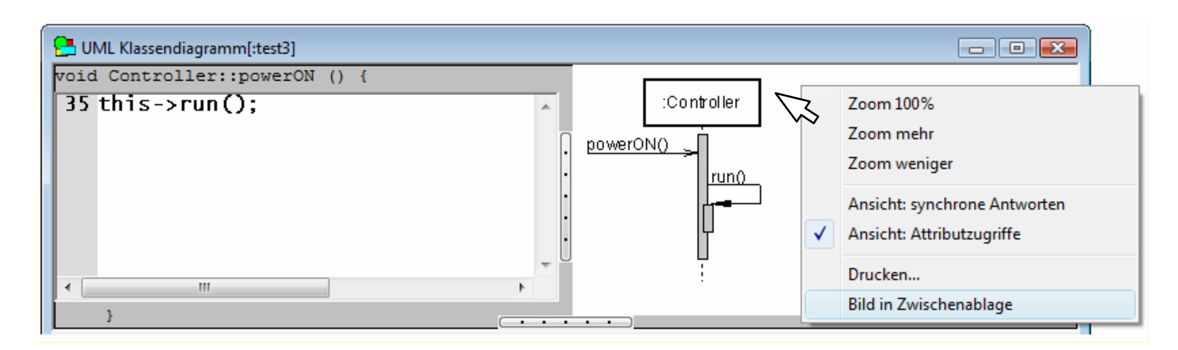


Nachricht an die aktuelle Instanz:



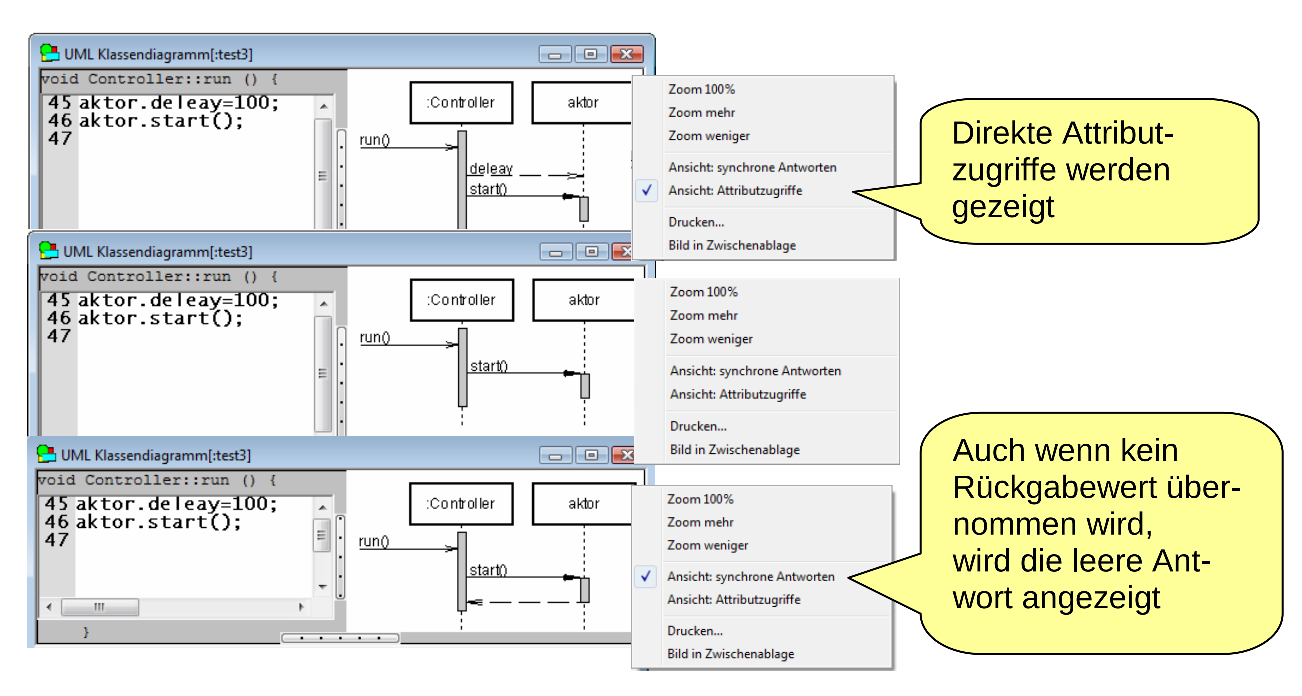
13.3 Sequenzen weiter verwenden

Sie können das automatisch erzeugte Sequenzdiagramm weiter verwenden, indem Sie es auf einen Drucker ausgeben oder die Darstellung in die Zwischenablage kopieren um diese zum Beispiel in eine Textverarbeitung einzufügen.



13.4 Besonderheiten des SiSy-Sequenzdiagramms

Für die direkten Zugriffe auf öffentliche Attribute einer Klasse trifft die UML derzeit keine verbindlichen Regeln. Es wird davon ausgegangen, dass Klassenattribute prinzipiell geschützt sind und nur über Methoden (Operationen) ein Zugriff erfolgen kann (Kapselung und Nachricht). Des Weiteren kann man davon ausgehen, dass auch eine leere (void) Antwort eine Antwort ist und angezeigt werden sollte. Dazu kann die Darstellung entsprechend über das Kontextmenü des Sequenzdiagramms angepasst werden.



14 Einstellungen Fuse- und Lock-Bits mit SiSy

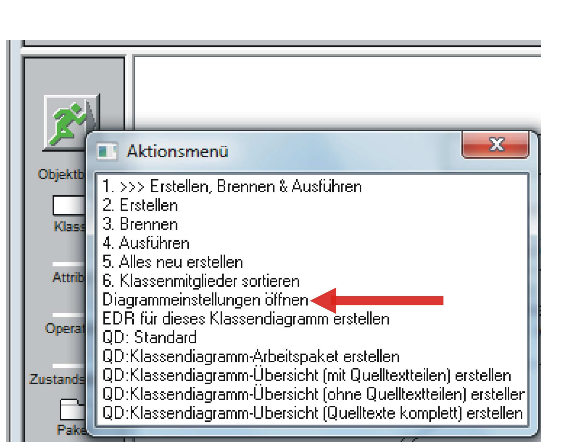
14.1 Einleitung

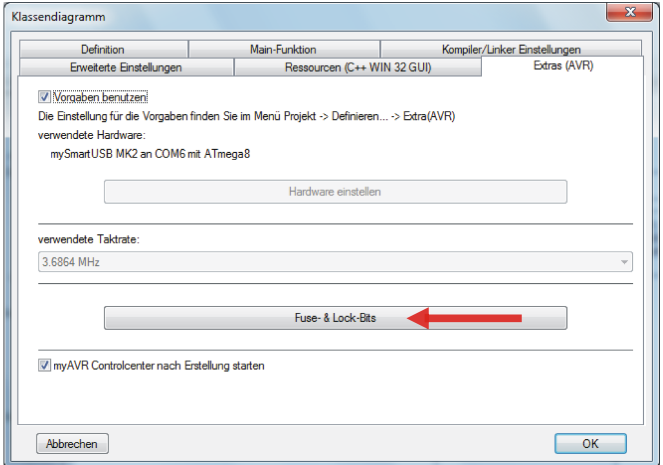
Fuse- und Lockbits (engl. fuse = Sicherung, engl. lock = Schloss) nennt man die Bits in bestimmten Registern des AVR zum Konfigurieren des Controllers. Die Fusebits müssen über ein entsprechendes Interface (Software) eingestellt werden. Der normale Programmiermodus verändert die Fuse- und Lockbits nicht. Je nach Controllertyp sind unterschiedliche Fuse- und Lockbits verfügbar. Die verbindliche und exakte Beschreibung findet man im jeweiligen Datenblatt des Controllers. Das falsche Setzen der Fuse- und Lockbits zählt zu den häufigsten Problemen bei der Programmierung von AVR-Controllern, daher sollte hier mit Umsicht vorgegangen werden. Das Verändern der Fusebits sollte man nicht als Anfänger vornehmen. Das geöffnete Datenblatt zur Überprüfung der Konfiguration ist das wichtigste Instrument um Fehler zu vermeiden.

14.2 Fuse- und Lockbits, Benutzeroberfläche in SiSy AVR

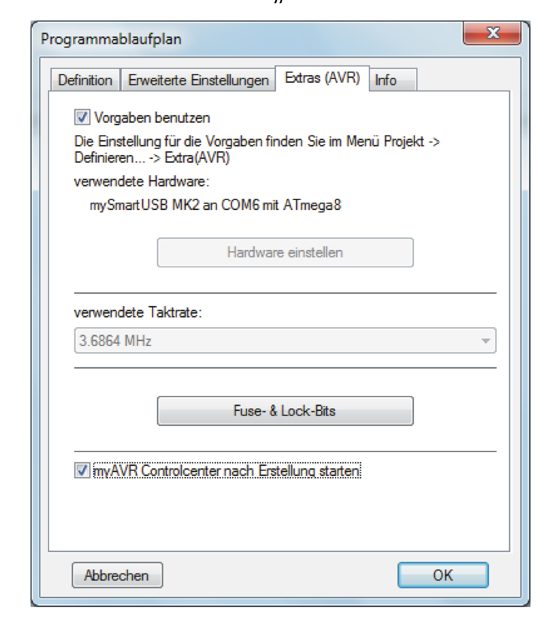
Sie erreichen die Benutzeroberfläche zum Auslesen, Verändern und Programmieren der Fuse- und Lockbits in SiSy AVR wie folgt:

1. Hauptmenü -> Werkzeuge -> myAVR ProgTool -> Fuse- und Lockbits

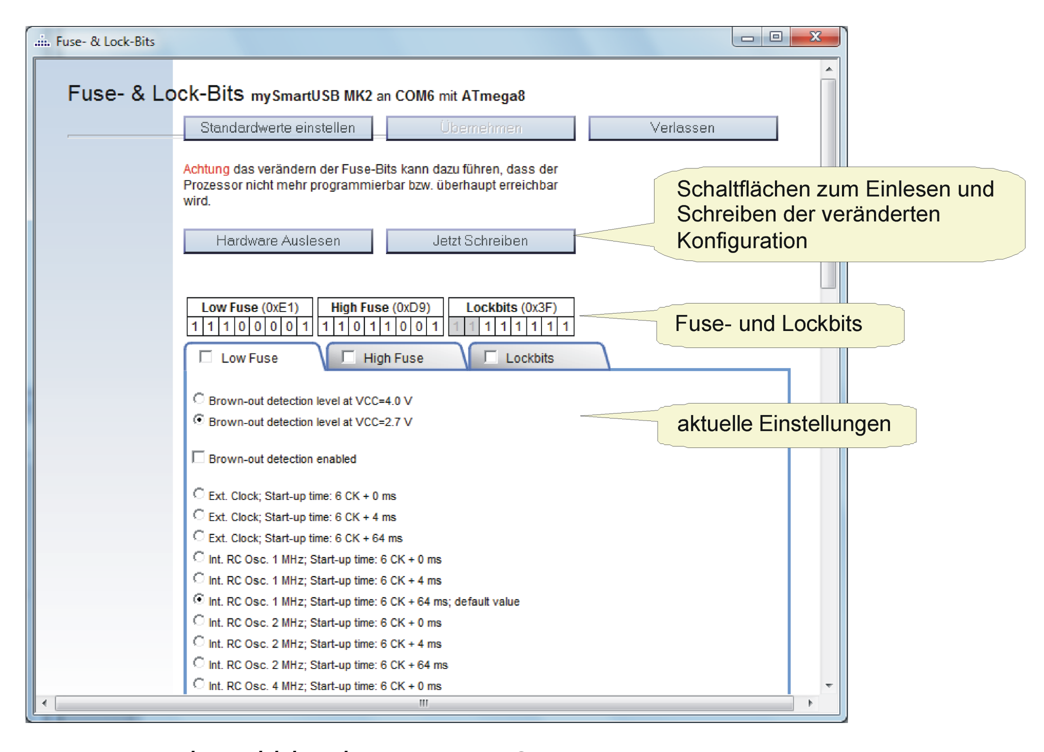




2. Auf Programm-Objekten wie „kleines Programm“, „Programm“, „PAP“ usw. rechte Maustaste -> Kontextmenü -> Definieren … Dialogfeld Extras (AVR) -> Schaltfläche „Fuse- & Lock-Bits“.

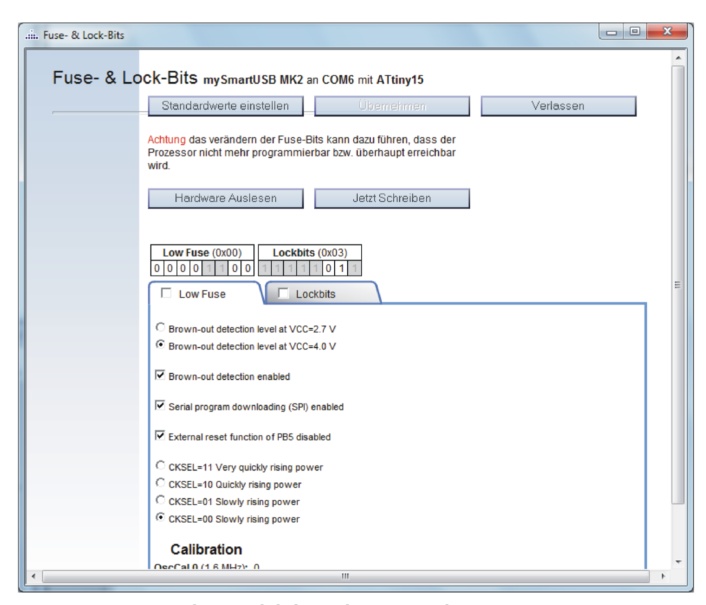


Beim Start der Fuse- und Lockbits-Benutzeroberfläche wird eine Verbindung zum Controller aufgebaut, der Controllertyp ermittelt, die Fuse- und Lockbit-Definitonen des Controllers geladen, die Fuse- und Lockbit Einstellungen des Controllers ausgelesen und angezeigt. Dieser Vorgang kann je nach Controllertyp und Verbindung einige Sekunden dauern. Die Verbindung zum Controller wird, solange diese Benutzeroberfläche offen ist, dauerhaft offen gehalten.



Fuse- und Lockbits des ATmega8

Die Benutzeroberfläche passt sich dem ermittelten Controller und den dazugehörigen Definitionsdaten automatisch an. Es werden immer nur die Optionen angezeigt, die zum ermittelten Controller gehören. Vergleichen Sie dazu immer das betreffende Datenblatt. Es ist nicht zulässig, den Programmer oder den Controller während der Sitzung zu entfernen oder zu wechseln. Dazu ist dieses Fenster zu schließen, danach der Controller oder Programmer zu wechseln und die Benutzeroberfläche erneut zu starten.



Fuse- und Lockbits des ATtiny15

14.3 Fuse- und Lockbits verändern

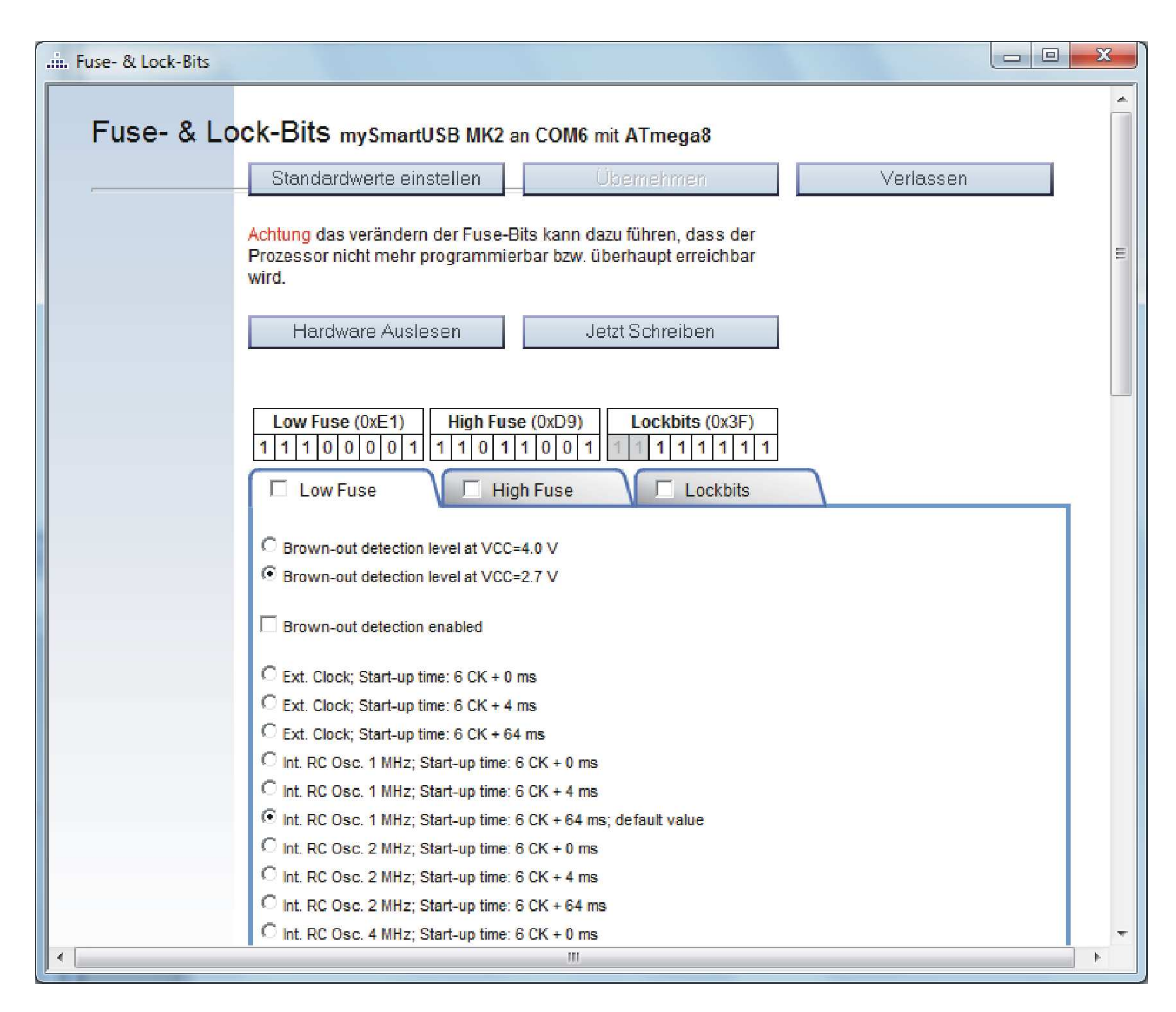
Zum Verändern der Fuse- und Lockbits sollte der entsprechende Abschnitt im Datenblatt des Controllers studiert werden. Über die Dialogfelder Low-, High- und Extended-Fuse- sowie Lockbits können die einzelnen Optionen bequem ausgewählt werden. Die Änderungen werden im Anzeigebereich für die Fuse- und Lockbits visualisiert. Erst mit dem Betätigen der Schaltfläche „Jetzt Schreiben“ werden die neuen Einstellungen an den Controller übertragen.

Beachte: Falsche Einstellungen der Fuse- oder Lockbits können dazu führen, dass der Controller in der aktuellen Hardware nicht mehr angesprochen werden kann.
Häufige Fehleinstellungen durch unerfahrene Entwickler sind:

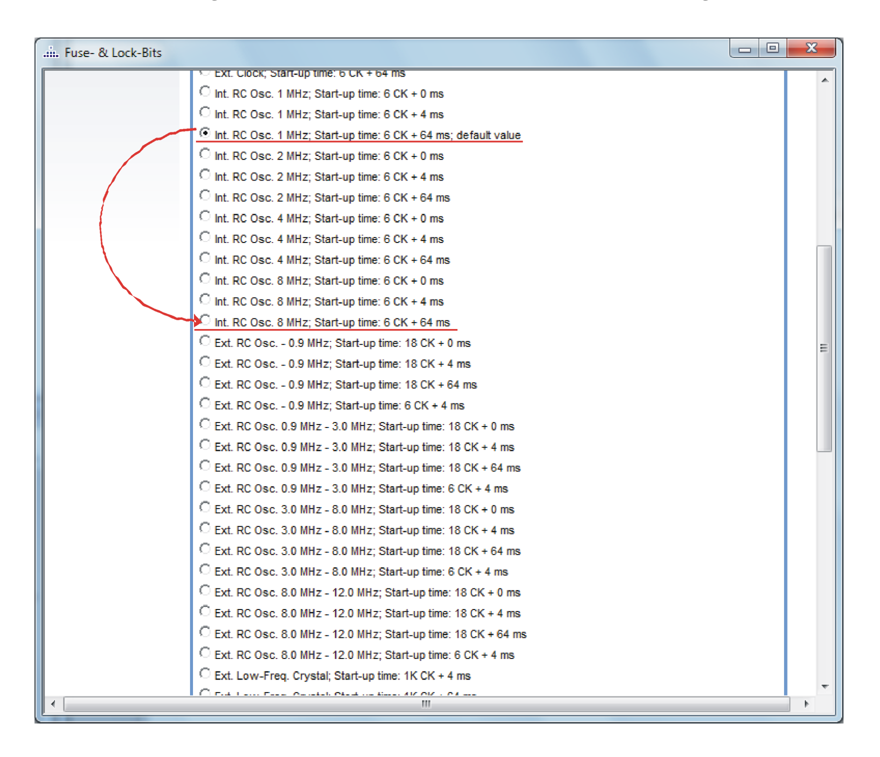
- **Reset disable ->** führt dazu, dass kein ISP mehr möglich ist
- **ISP enable ausgeschaltet ->** führt dazu, dass kein ISP mehr möglich ist
- **Taktquelle umgeschaltet ->** führt u.U. dazu, dass der Controller nicht arbeitet

Im Folgenden wird die Vorgehensweise beschrieben, wie die Taktquelle eines ATmega8 vom internen 1 MHz Oszillator auf intern 8 MHz umgeschaltet wird.

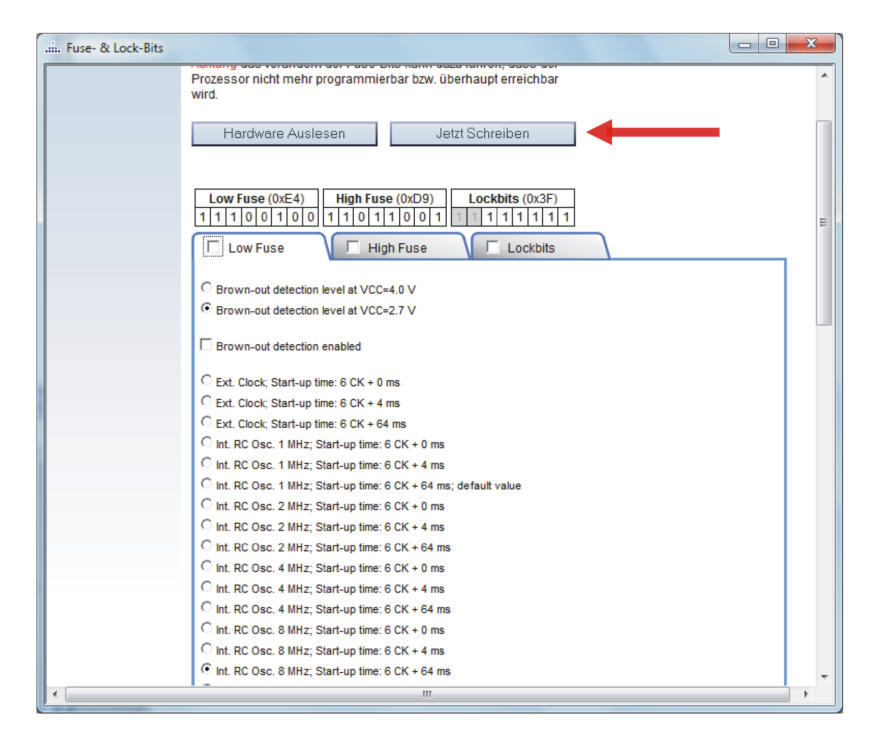
1. Das Board und den Programmer anschießen.
2. Benutzeroberfläche für das Verändern der Fuse- & Lockbits starten.



3. In der Liste der Optionen nach unten scrollen und **`Int. RC. Osc. 8MHz`** auswählen. Vergleichen Sie dazu die Beschreibung im Datenblatt des ATmega8.



4. Die Optionsseite wieder zurückscrollen und die veränderten Fusebits überprüfen.



5. Die Schaltfläche „Jetzt Schreiben“ wählen und die Sicherheitsabfrage bestätigen. Danach sollten die Einstellungen überprüft werden durch das Betätigen der Schaltfläche „Hardware Auslesen“.

Über die Schaltfläche „Verlassen“ kann die Sitzung zum Verändern der Fuse- und Lockbits beendet werden. Die Verbindung zum Controller und Programmer wird dann geschlossen.

15 Entwicklung eines Windows-Programms mit SiSy AVR++

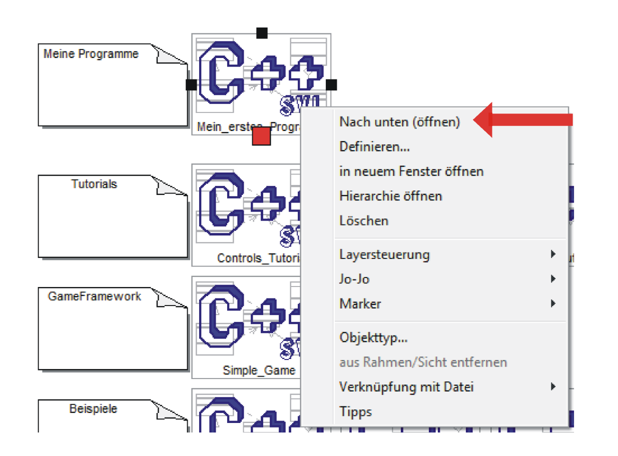
Das nachfolgende Programmbeispiel kann nur nachgearbeitet werden, wenn in Ihrer SiSy-Ausgabe das Add-On „SVL" enthalten ist. In Sisy AVR++ ist es standardmäßig integriert.
Die Funktion des nachfolgenden Programms ist es, ein Fenster mit zwei Schaltflächen zu erstellen, welche das Fenster schließen bzw. eine Message Box erzeugen.

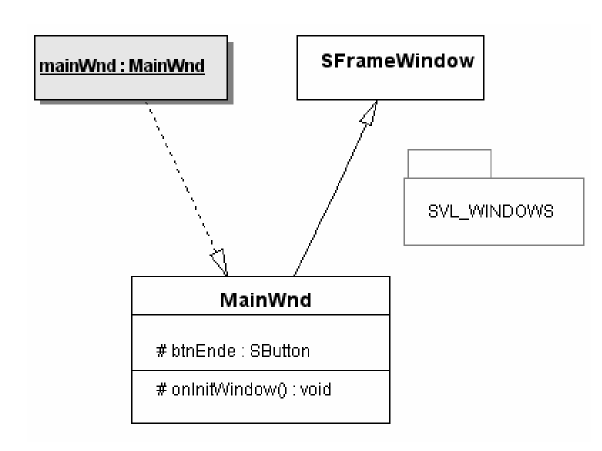
15.1 Ein neues Projekt anlegen

Starten Sie SiSy und wählen Sie „neues Projekt erstellen". Vergeben Sie den Projektnamen „SVL_Button" und bestätigen Sie mit *OK*. Wählen Sie nun das Vorgehensmodell „UML mit SVL (Smart Visual Library)" und laden Sie als Diagrammvorlage „Smart Visual Library (SVL)", welches neben Beispielen und Tutorials auch ein Grundgerüst für SVL-Anwendungen beinhaltet.

15.2 Grundgerüst für Fenster auswählen

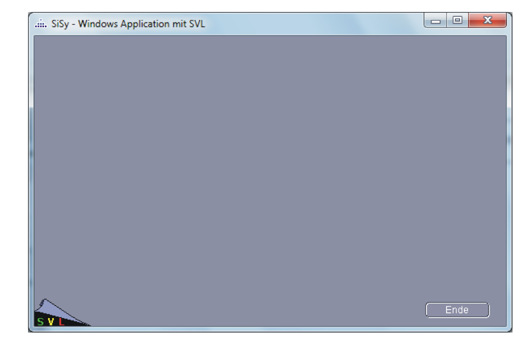
Sie sehen eine Auswahl an Tutorials und Beispielen sowie „Meine Programme". Wählen Sie das Klassendiagramm „Mein_erstes_Programm" mit der rechten Maustaste aus und klicken Sie auf „nach unten (öffnen)". Sie sehen das verfeinerte Klassendiagramm, welches das Grundgerüst für Ihr Programm ist.





Mit der Aktion „Erstellen & Ausführen", welche Sie über das grüne Ampelmännchen erreichen, wird der dazugehörige Quellcode generiert, das Programm kompiliert und ausgeführt. Als Ergebnis erhalten Sie ein Fenster, welches bereits eine Schaltfläche für das Schließen des Fensters besitzt.

Diesem Fenster kann man nun beliebig viele Controls, d.h. Elemente (Button, Checkbox, Textfeld etc.), hinzufügen und diesen einzelnen Controls entsprechende Funktionen zuweisen.
Dies erreicht man über eine direkte Eingabe des Quelltextes oder über den integrierten Control Wizard, welcher das schnelle Einfügen und Konfigurieren der Controls ermöglicht und Bestandteil sowie Arbeitsmittel in der Abarbeitung dieses Beispiels ist.

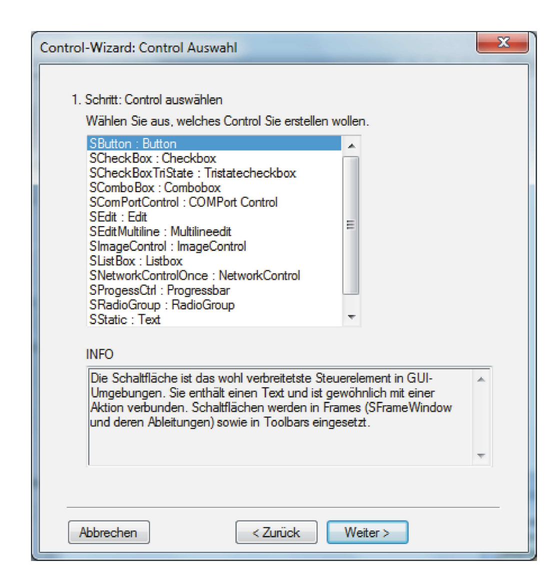


15.3 Schaltfläche mit Hilfe des Control Wizards erstellen

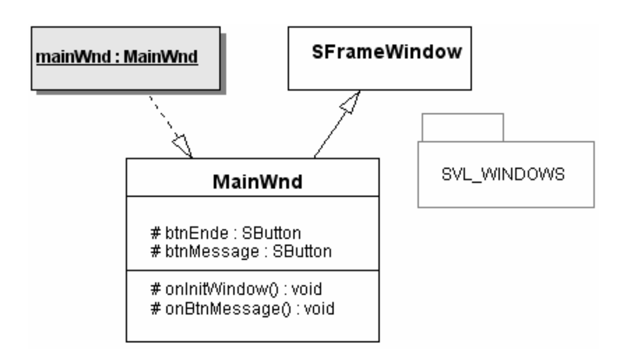
Fügen Sie nun dem Fenster eine weitere Schaltfläche hinzu. Dazu wählen Sie aus der Objektbibliothek den Typ „Attribut“ und ziehen ihn per Drag & Drop Verfahren in die Klasse „MainWnd“.

Daraufhin öffnet sich automatisch der Control Wizard, welcher das Anlegen von Controls und Funktionen erleichtert indem er Schritt für Schritt alle notwendigen Anweisungen ausführt.

Wählen Sie im ersten Schritt „SButton : SButton“ und klicken Sie auf „Weiter“ um eine Schaltfläche zu erzeugen.

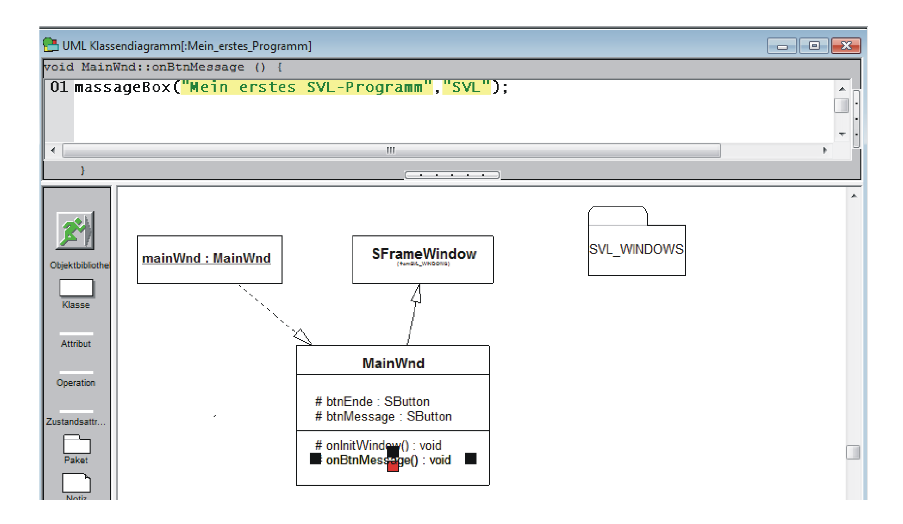


Als nächstes wählen und geben Sie als Beschriftung des Elements den Namen „Message“ ein. Der Variablenname wird dabei automatisch vergeben. Klicken Sie nun „Weiter“ um zum nächsten Schritt zu gelangen, welcher sich mit der Positionierung der Schaltfläche beschäftigt. Diesen und den Folgeschritt (Tooltipp) überspringen Sie, indem Sie dreimal auf „Weiter“ und dann auf „Fertig stellen“ klicken. Die Schaltfläche und die Funktion OnButtonMessage() wird nun erstellt und der entsprechende Quellcode, welcher zur Generierung notwendig ist, wird automatisch erzeugt.



15.4 Quellcode hinzufügen

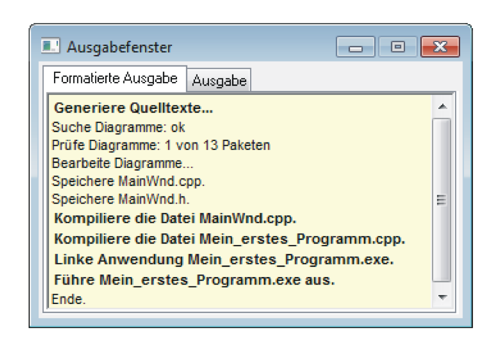
Damit die neu erzeugte Schaltfläche per Mausklick eine MessageBox aufruft, müssen Sie der Funktion OnBtnMessage() noch den entsprechenden Quellcode hinzufügen. Dazu wählen Sie in der Klasse „MainWnd“ die Funktion OnBtnMessage() aus und fügen nun folgenden Quellcode ein:



messageBox("Mein erstes SVL-Programm","SVL");

15.5 Kompilieren und Linken des fertigen Programms

Klicken Sie auf das grüne Ampelmännchen und wählen Sie „>>> Erstellen & Ausführen“ um das Projekt zu kompilieren und zu linken. Das Programm wird nach erfolgreichem Abschließen des Vorgangs automatisch gestartet.



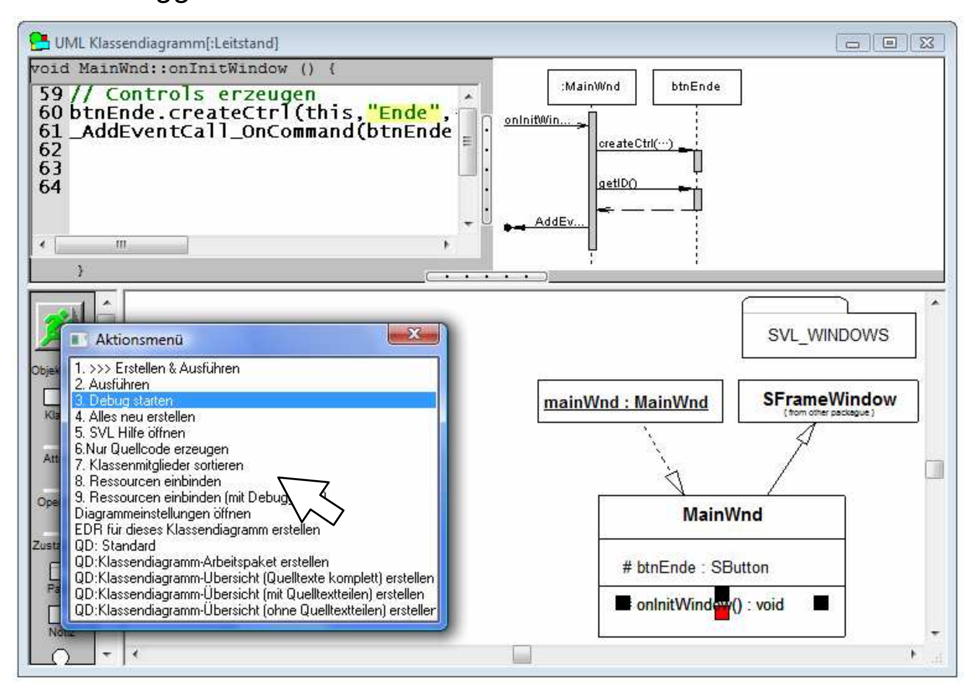
Sollten Sie keine Fehler im Ausgabefenster erhalten haben, dann können Sie nun durch einen Klick auf die Schaltfläche „Message“ eine MessageBox mit der Nachricht „Mein erstes SVL-Programm“ erzeugen.
Für eine andere Positionierung der Schaltfläche klicken Sie in der linken unteren Ecke auf das SVL-Logo und wählen „Positionierung“. Nun können Sie per Drag & Drop Verfahren die einzelnen Elemente des Fensters neu positionieren.



15.6 Debugging von SVL Programmen

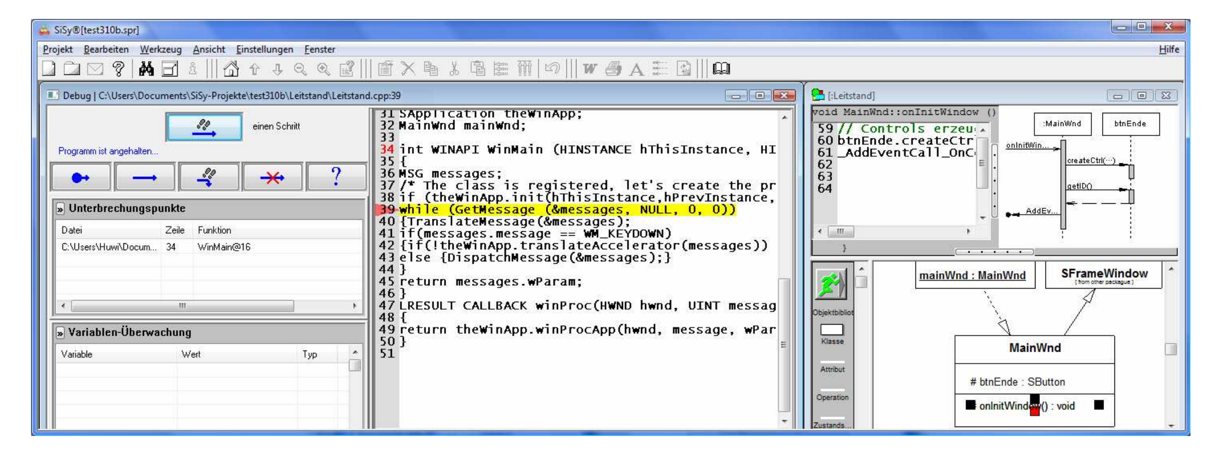
SiSy verfügt über ein Debug-Interface um C oder C++ Programme zu debuggen. Unter Debugging versteht man bestimmte Arbeitstechniken zur Fehlersuche in laufenden Programmen. Dazu gehören vor allem der Schrittbetrieb, Unterbrechungspunkte, Kontrollausgaben und Datenüberwachung. Der integrierte Debugger ermöglicht es Ihnen, sowohl auf Quellcodeebene als auch auf Modellebene Fehler zu suchen. Das Starten des Debuggers erfolgt in der Regel über das Aktionsmenü des aktiven Diagramms.

Den Debugger starten:



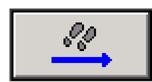
Der Debugger startet und wird mit dem dazugehörigen Quelltext angezeigt. Dabei wird in jedem Fall ein Unterbrechungspunkt beim Programmstart (main / WinMain) gesetzt und das Programm dort angehalten. Jetzt kann der Anwender weitere Unterbrechungspunkte hinzufügen. Die Position an der ein Programm angehalten wurde, ist im Quelltext als Zeile markiert. Wenn die entsprechende Position im Modell ermittelt werden konnte, wird das Modellfenster nachgeführt und das betreffende Element selektiert.

SiSy mit geöffneten Debugger im Klassendiagramm:



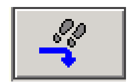
Unterbrechungspunkte können im Quellcodeeditor des Diagrammfensters hinzugefügt werden. Nutzen Sie dazu den Menüpunkt *Breakpoint umschalten* im Kontextmenü des Editors.

Die Schaltflächen des Debuggers:



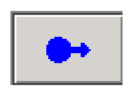
step (over)

einen Schritt: dient dazu, das Programm Schritt für Schritt ablaufen zu lassen. Dies hilft vor allem bei der Fehlersuche, da hier Zeile für Zeile des Quellcodes entsprechend des Programmablaufs abgearbeitet wird. Funktionen werden dabei jedoch als ganze Einheit betrachtet.



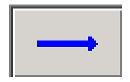
step in to

einen Schritt hinein: verzweigt das Programm zusätzlich in die Einzelanweisungen von aufgerufenen Funktionen.



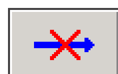
restart

Programm neu starten



go

Programm laufen lassen



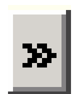
end

Debugger beenden; kann auch erfolgen, wenn das Programm nicht komplett abgearbeitet wurde.



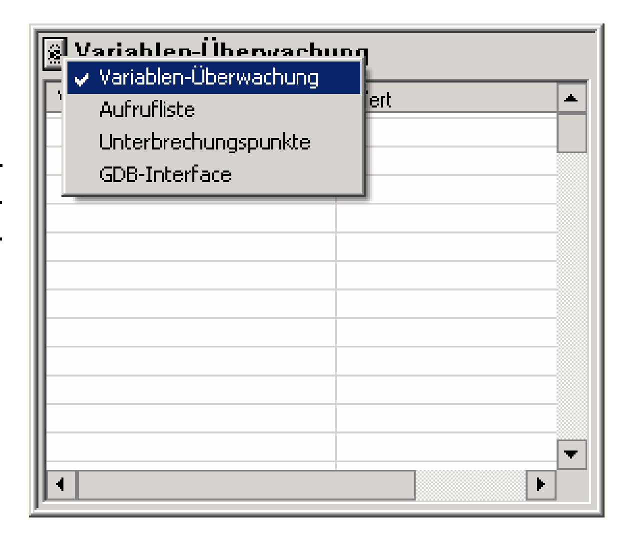
help

Hilfe zu Debugger öffnen: hier werden weitere Hinweise zum Debugger gegeben.



menu

Auswahl-Menü öffnen: ein Drop-Down Menü wird geöffnet, aus dem wie in der Abbildung dargestellt, verschiedene Debug-Ansichten gewählt werden können.



16 Programmieren mit dem UML Zustandsdiagramm

16.1 Einführung in die Zustandsmodellierung

Viele Problemstellungen in der Softwareentwicklung lassen sich als eine Folge von Zuständen und Zustandsübergängen (Zustandsautomat, state machine) verstehen und auch als solche implementieren. Dabei handelt es sich im Quellcode um oft recht aufwendige Fallunterscheidungen bzw. if/else Konstruktionen. SiSy ist mit dem entsprechenden Add-On in der Lage, aus UML-Zustandsdiagrammen den entsprechenden Quellcode automatisch zu generieren.

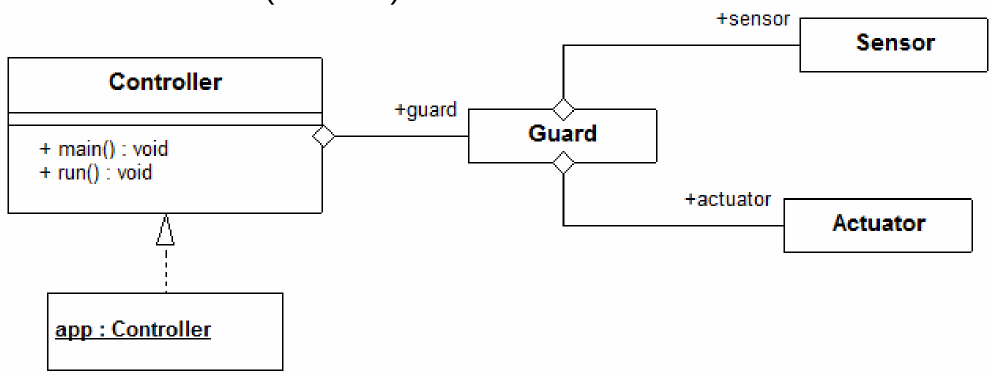
Notationsübersicht Zustandsdiagramm (Auszug)

Notation	Bezeichnung	Beschreibung
	Startknoten	Ein Startknoten aktiviert Abläufe. Startknoten besitzen nur ausgehende Kanten.
Zustand entry / wird beim Betreten ausgeführt do / wird wiederholt während des Zusandes ausgeführt event / wird bei einem bestimmten Ereignis ausgeführt exit / wird beim Verlassen ausgeführt	Zustand	Zustände eines Objektes sind gekennzeichnet durch zustandsspezifische Aktivitäten beim Einnehmen, während und beim Verlassen des Zustandes (entry, do, exit)
	Zustandsübergang	Ein Zustandsübergang (transition) repräsentiert mindestens zwei Aktivitäten. Eine Aktivität vom Typ „exit“ beim alten und eine Aktivität vom Typ „entry“ beim neuen Zustand.
	Endknoten	Ein Endknoten beendet Abläufe und besitzt nur eingehende Kanten.

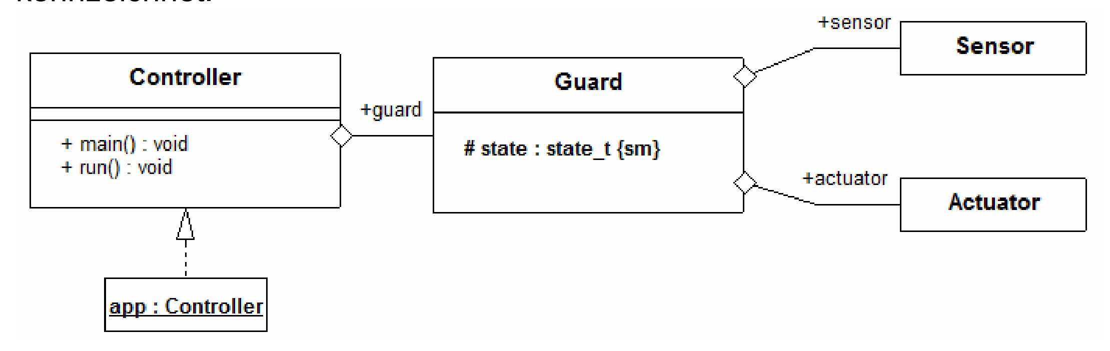
16.2 Erstellen von Zustandsdiagrammen

Das Zustandsdiagramm mit Quellcodegenerierung erhalten Sie über ein spezielles Attribut im Klassendiagramm der UML. Dieses Zustandsattribut kann in mit einem Zustandsdiagramm verfeinert werden. Dabei wird der Quellcode in spezielle Klassenmethoden generiert. Der Entwickler kann selbst Methoden definieren, die an der Verarbeitungslogik des Zustandsautomaten (state machine) beteiligt sind. Voraussetzung ist es, ein neues Projekt mit einem Klassendiagramm anzulegen.

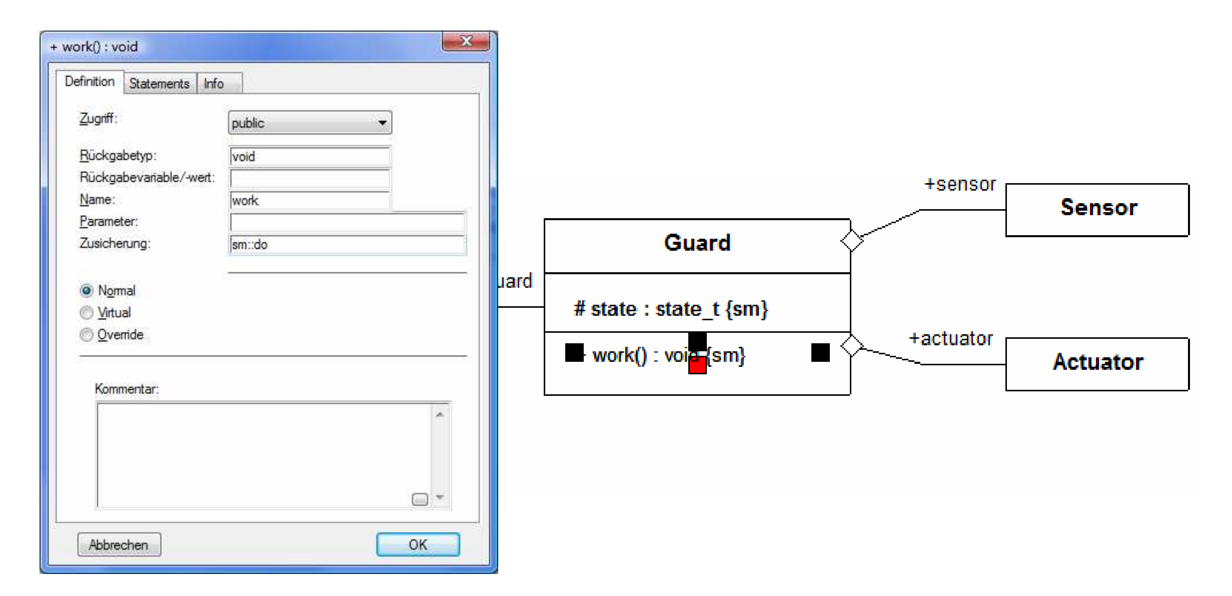
Die Aufgabe welche mit einem Zustandsdiagramm gelöst werden soll lautet wie folgt: Ein Controller steuert einen Wächter (guard) der einen Sensor überwacht und entsprechend einen Aktor (actuator) ansteuert.



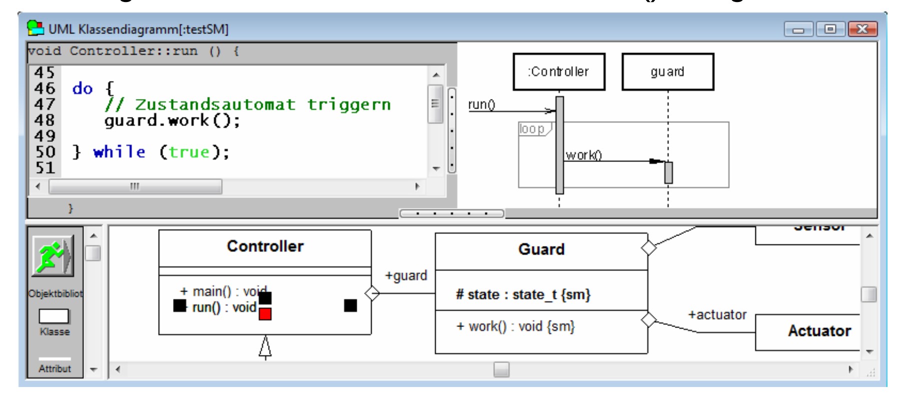
Der Wächter (guard) ist ein Zustandsautomat (state machine). Somit ist der Klasse „Guard“ ein Zustandsattribut zuzuordnen. Dazu muss ein Objekt vom Typ *Zustandsattribut* per Drag und Drop aus der Objektbibliothek in die betreffende Klasse gezogen werden. Die Zugehörigkeit zum Zustandsdiagramm wird mit der Zusicherung {sm} gekennzeichnet.



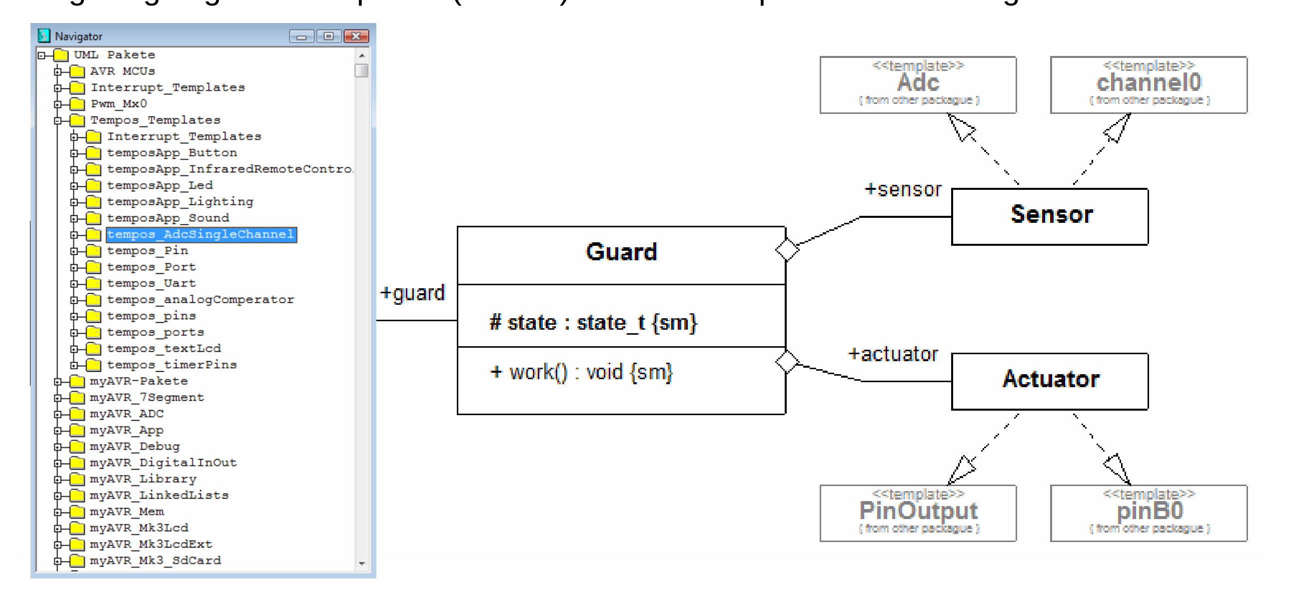
Für die Verarbeitungslogik des Zustandsautomaten soll eine eigene Methode angelegt werden, welche in der Hauptschleife der Mikrocontrolleranwendung (mainloop) zyklisch aufgerufen wird. Damit die Methode im Zustandsdiagramm verfügbar ist, muss diese mit einer entsprechenden Zusicherung versehen werden (z.B.: {sm::do}). Dazu muss aus der Objektbibliothek per Drag und Drop eine Operation (Methode) in die betreffende Klasse gezogen und wie folgend dargestellt definiert werden.



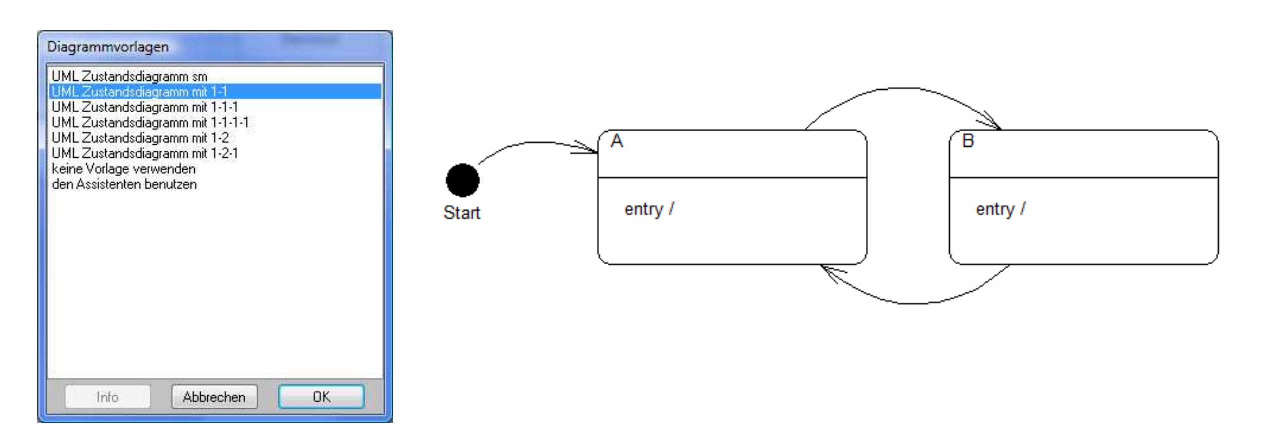
Diese Methode benötigt keine Statements. Der Codegenerator fügt beim Bilden der Anwendung hier Code entsprechend der Modellierung im Zustandsdiagramm ein. Es ist jedoch nötig, die Verarbeitungsfunktion kontinuierlich aufzurufen um die Zustandswechsel auszulösen (trigger). Das erfolgt in diesem Fall in der Methode *run*(). Es gibt in den SiSy-Klassenbibliotheken spezielle Klassen, die bereits einen Zustandsautomaten abbilden. Wird von solchen Basisklassen abgeleitet, sind die Auslöser (trigger) für den Zustandsautomaten bereits implementiert.



Für eine schnelle Realisierung der Klassen „Sensor“ und „Actuator“ wird im folgenden Beispiel das myAVR Programmiermodell *Tempos* benutzt. Dabei müssen den Klassen lediglich geeignete Templates (Muster) aus der Tempos-Bibliothek zugewiesen werden.

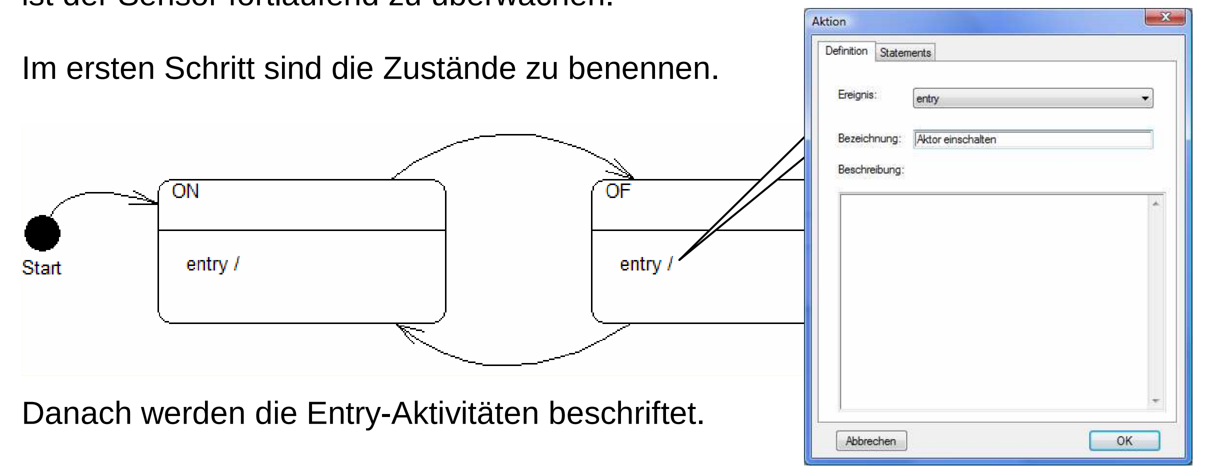


Die Modellierung der Zustände und Zustandswechsel erfolgt in Verfeinerung des Zustandsattributes. Dazu selektieren Sie das Zustandsattribut und wählen im Kontextmenü den Menüpunkt *nach unten (öffnen)*. Dabei wird bereits eine Auswahl typischer Zustandsvarianten als Diagrammvorlage angeboten. Wählen Sie die Vorlage 1-1.

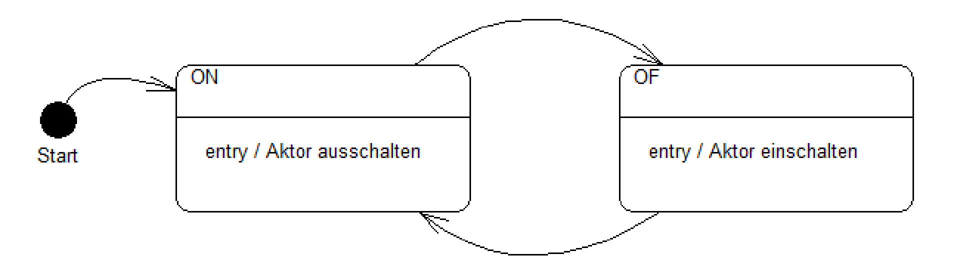


Die Zustände des Wächters sollen ON und OFF sein. Dabei wird nach dem Start der Zustand OFF eingenommen. Die Zustandswechsel erfolgen in Abhängigkeit der Sensorwerte. Wird der Zustand OFF eingenommen, muss der Aktor ausgeschaltet werden, wird der Zustand ON eingenommen, ist der Aktor einzuschalten. Während der Zustände ist der Sensor fortlaufend zu überwachen.

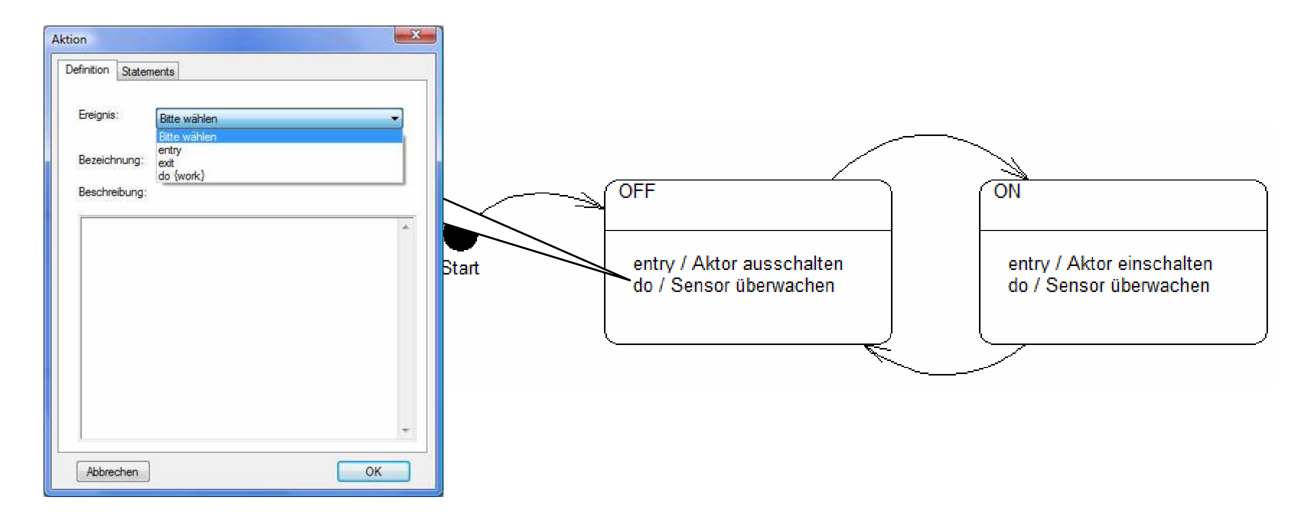
Im ersten Schritt sind die Zustände zu benennen.



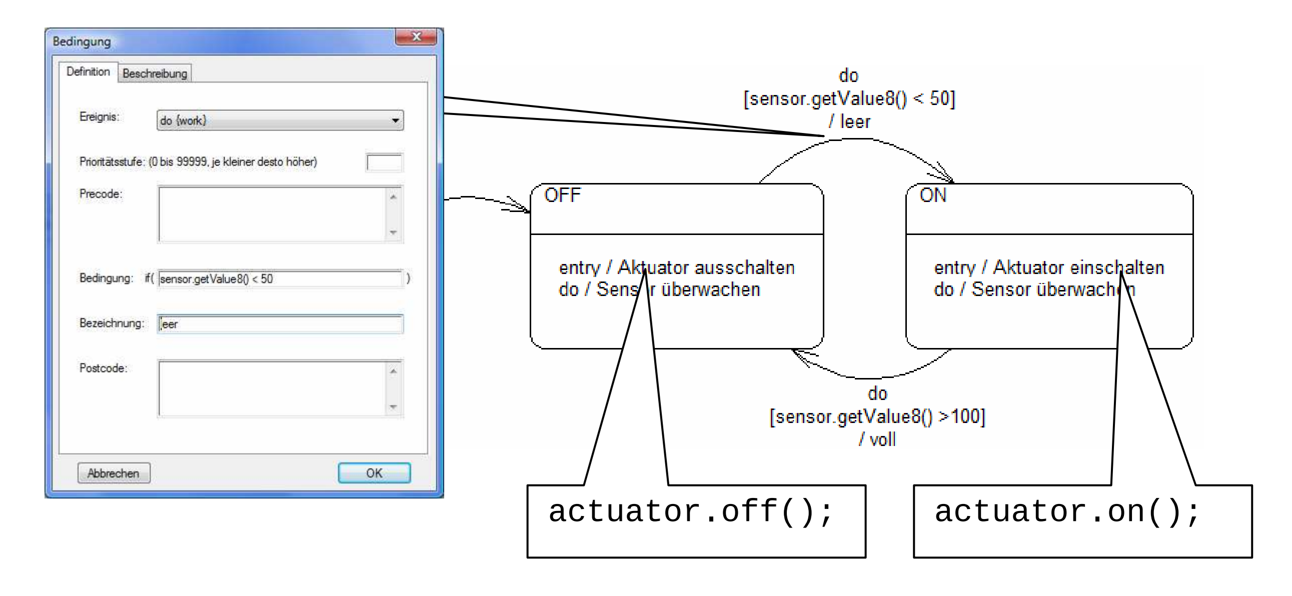
Danach werden die Entry-Aktivitäten beschriftet.



Um weitere Aktionen (entry/do/exit) einzufügen, müssen per Drag und Drop Objekte vom Typ Aktion in die betreffenden Zustände gezogen werden. Das Ereignis/Aktivitätstyp ist auszuwählen und die Aktivität zu beschriften.



Die Zustandsübergänge müssen definiert werden indem festgelegt wird, wann der Zustandswechsel erfolgen darf und unter welchen Bedingungen. Des Weiteren sollte der Zustandsübergang eine Bezeichnung erhalten.



Ist das Zustandsmodell erarbeitet, können die nötigen Statements für den Programmcode eingearbeitet werden. Dazu wird die betreffende Aktivität selektiert und im Quellcodeeditor die entsprechenden Befehle eingegeben. Danach kann im dazugehörigen Klassendiagramm (rechte Maustaste Menüpunkt *nach oben*) das Programm gebildet, übertragen und getestet werden.

17 Weitere Informationen zu SiSy

Versionen und Ausgaben Stand Mai 2011

Die Laufzeitkomponenten von SiSy, das Metamodell, Add-On's sowie spezifische Skripte, Dokumente und Daten sind lizenzpflichtige Produkte. Ausgenommen sind zusätzlich bzw. nachträglich installierte Komponenten Dritter, die von SiSy als Modellierungswerkzeug lediglich zum Beispiel über Skripte angesprochen werden oder in die sich SiSy über offengelegte Schnittstellen eingebettet hat. Solche Produkte unterliegen den Lizenz und Nutzungsbedingungen des jeweiligen Herstellers bzw. Herausgebers.

Ausgaben

- SiSy Professional
  umfasst alle verfügbaren Add-Ons
- SiSy Business
  umfasst Add-Ons für Prozessmodellierung, Qualitäts- und Projektmanagement
- SiSy Developer
  umfasst umfangreiche Add-Ons zur Systementwicklung
- SiSy AVR ++
  umfasst die Add-Ons AVR, ausgewählte Teile der UML, SVL und SysML
- SiSy AVR
  beinhaltet nur das Add-On AVR

Lizenzmodelle

- Evaluation License
  erlaubt die Nutzung auf einem Arbeitsplatzrechner für Evaluierungszwecke
  Lizenzlaufzeit 1 Jahr
- Education License
  erlaubt die Nutzung auf einem Arbeitsplatzrechner für Ausbildungszwecke
  Vorraussetzung: Bildungseinrichtung, Student, Schüler, keine kommerzielle Nutzung
- Private License
  erlaubt die Nutzung auf einem Arbeitsplatz für private oder hausinterne Zwecke
  Vorraussetzung: ausschließlich für private, nicht kommerzielle Nutzung
  in Unternehmen hausinterne Nutzung der mit SiSy erstellten Produkte
- Single License
  erlaubt die Nutzung auf einem Arbeitsplatz kommerzielle Zwecke oder
  in öffentlichen Verwaltungen
- Enterprise License
  erlaubt die Nutzung auf bis zu 20 Arbeitsplatzen für kommerzielle Zwecke oder
  in öffentlichen Verwaltungen

Ausgabe	Enterprise	Single	Private	Education	Evaluation
SiSy Professional	25.000,- €	2800,- €	990,- €	590,- €	Anfrage
SiSy Business	20.000,- €	2300,- €	890,- €	399,- €	Anfrage
SiSy Developer	9900,- €	890,- €	499,- €	199,- €	Anfrage
SiSy AVR ++	4900,- €	399,- €	199,- €	99,- €	Anfrage
SiSy AVR	1900,- €	299,- €	99,- €	49,- €	Anfrage

Anhang: Tastaturbelegung, allgemein

Die Tastenbelegung ist abhängig vom jeweiligen Diagramm und der verwendeten Ausgabe:

F1	Hilfe wird geöffnet
F2	Zoomen
F4	Textfeld / Infofeld vergrößern
F5	Farbe bei Rahmen wird geändert; Form am Anfang einer Verbindung ändert sich
F6	Bei Rahmen und Verbindungen ändert sich die Form
F7	Bei Rahmen und Verbindungen ändert sich der Linientyp
F8	Form am Ende einer Verbindung ändert sich
F9	Bewirkt, dass der Mittelpunkt einer Kante auf Null gesetzt wird
ESC	Im Diagramm: Nach oben
	Im Dialog: Bricht ihn ab ohne zu speichern
Tab	In der Reihenfolge, in der die Objekte erstellt wurden, werden sie markiert
Umschalttaste + Enter	Objektbeschreibung wird geöffnet
Leertaste	Objektbeschreibung wird geöffnet
Alt + Enter	Dialog Definieren
Strg + Enter	Diagramm nach unten
Strg + `R	Report für selektiertes Objekt
Strg + `A`	Executebefehl ausführen (nur in bestimmten Ausgaben)
Strg + `D`	Diagrammreport wird aufgerufen
Strg + `I`	Import von Diagrammen
Strg + `T`	Tokensteuerung starten/beenden
Strg + `X`	Export von Diagrammen
Strg + `+`	Selektiertes Objekt wird vergrößert (nur in bestimmten Ausgaben)
Strg + `-`	Selektiertes Objekt wird verkleinert (nur in bestimmten Ausgaben)
Strg + `*`	Ursprüngliche Objektgröße wird wiederhergestellt

Strg + Maustaste	Selektiertes Objekt wird innerhalb des Diagramms kopiert
Strg + Cursortasten:	
- Cursor nach links	Selektiertes Objekt wird in X-Richtung verkleinert
- Cursor nach rechts	Selektiertes Objekt wird in X-Richtung vergrößert
- Cursor nach oben	Selektiertes Objekt wird in Y-Richtung vergrößert
- Cursor nach unten	Selektiertes Objekt wird in Y-Richtung verkleinert
Enter	Editormodus zum Definieren der Objekte
Entf	Löschen
Cursortasten	Selektiertes Objekt wird verschoben (in Verbindung mit der Umschalttaste sind größere Schritte möglich)
\`+\`	Diagramm vergrößern
\`-\`	Diagramm verkleinern
\`*\`	Einpassen des Diagramms

Anhang: Mausoperationen

Die Maus hat in SiSy eine Anzahl von nützlichen Funktionen, welche die Arbeit in Projekten erleichtern.

Selektion	*Klick auf Objekt* Objekt ist markiert und kann separat weiterbearbeitet werden.
Selektion aufheben	*Klick auf Fensterhintergrund* Aufhebung der Objektmarkierung.
Mehrfachselektion	*Umschalttaste + Klick auf Objekt* Selektion/Markierung von mehreren Objekten zur Weiterbearbeitung.
	Markise Mit *Shift* und *gedrückter linker Maustaste* auf Fensterhintergrund und Ziehen eines Rechtecks über zu markierende Objekte.
Verschieben	*Drag & Drop im Diagramm* Objekt mit linker Maustaste anfassen und verschieben. Objekte werden am Raster verschoben.
	Umschalttaste + Drag & Drop im Diagramm Verschieben von Objekten ohne Raster.
Fensterinhalt schieben	*Linke und rechte Maustaste drücken + Verschieben der Maus im Diagramm* Der komplette Diagramminhalt wird geschoben.
Objekt kopieren	*STRG + Drag & Drop* Maustaste gedrückt halten und Mauszeiger vom Objekt auf den Fensterhintergrund führen. Eine Originalkopie des Objektes wird im aktuellen oder in einem anderen Diagramm erzeugt.
Referenz erzeugen	*Drag & Drop aus Navigator* Ziehen des gewünschten Objektes aus dem Navigator in das Diagramm. Es wird eine Referenz des gewählten Objektes erzeugt.
	Drag & Drop aus Objektbibliothek Rot beschriftete Objekte können nur als Referenz erzeugt werden. Eine Liste zur Auswahl des gewünschten Typs erscheint.
	Strg + Drag & Drop aus Objektbibliothek Eine Liste zur Auswahl der gewünschten Referenz des Originalobjektes erscheint.

	Drag & Drop aus anderem Diagramm Ziehen des gewünschten Objektes aus dem Quelldiagramm in das Zieldiagramm. Es wird eine Referenz des gewählten Objektes erzeugt.
Objekt anlegen	*Drag & Drop aus Objektbibliothek* Ein Objekt aus der Objektbibliothek wird im Diagramm angelegt und steht zur Verfeinerung bereit.
Objekt anhängen	*Drag & Drop Verteiler auf Fensterhintergrund* Durch Ziehen einer Kante vom Verteiler auf den Fensterhintergrund wird ein neues Objekt erzeugt. Nach Auswahl des Objekttyps sind die Objekte miteinander verbunden.
Objekte verbinden	*Drag & Drop Verteiler zu Objekt* Klick auf den Verteiler des zu verbindenden Objektes. Bei gedrückter linker Maustaste auf das gewählte Objekt ziehen. *Verbindung aus Objektbibliothek (in der UML)* Hierbei wird erst die gewünschte Verbindung in der Objektbibliothek angeklickt und danach die beiden zu verbindenden Objekte im Diagramm nacheinander.
Verbindung anordnen	*Drag & Drop Mittelpunkt* Beliebige Gestaltung der Verbindung durch Ziehen mit der Maus.
Verbindung ändern	*Drag & Drop Anfangs-/Endpunkt einer Kante* Für die Verbindung wird ein neues Zielobjekt gewählt.
Objekt definieren	*Doppelklick auf Objekt* Durch Doppelklick auf Objekte öffnet sich das Kontextmenü. Bei Abschalten des Menüs unter *Einstellungen*/*Menü bei Doppelklick* erscheint eine Zeile zur Namensgebung. Mit ESC wird die Eingabe bestätigt. *Doppelklick auf Verteiler* Es wird der Definieren-Dialog aufgerufen, in dem das Objekt benannt und beschrieben werden kann.
Kontextmenü öffnen	*Klick mit rechter Maustaste auf Objekt*
Fenster neu zeichnen	*Doppelklick auf Fensterhintergrund* *Hinweis*: Doppelklick mit linker Maustaste wirkt wie Enter.
Fenster aktualisieren	*Strg + Doppelklick auf Fensterhintergrund* Die vom Programm ausgeführten aber noch nicht sichtbargemachten Befehle werden im Fenster erstellt. Das Fenster wird aktualisiert.

